2013

Multiple Alignment of Protein Interaction Networks by Three-Index Assignment Algorithm

Arushi Arora
Universty of Windsor

Multiple Alignment of Protein Interaction Networks by Three-Index Assignment Algorithm

by

Arushi Arora

A Thesis

Submitted to the Faculty of Graduate Studies through Computer Science

in Partial Fulfillment of the Requirements for the Degree of Master of Science at the

University of Windsor

Windsor, Ontario, Canada

2013

Multiple Alignment of Protein Interaction Networks by Three-Index Assignment Algorithm

by

Arushi Arora

APPROVED BY:

_____

K. Tepe
Department of Electrical and Computer Engineering

_____

L. Rueda
School of Computer Science

_____

A. Ngom, Advisor
School of Computer Science

September 17, 2013

DECLARATION OF ORIGINALITY

I hereby certify that I am the sole author of this thesis and that no part of this thesis has been published or submitted for publication.

I certify that, to the best of my knowledge, my thesis does not infringe upon anyone's copyright nor violate any proprietary rights and that any ideas, techniques, quotations, or any other material from the work of other people included in my thesis, published or otherwise, are fully acknowledged in accordance with the standard referencing practices. Furthermore, to the extent that I have included copyrighted material that surpasses the bounds of fair dealing within the meaning of the Canada Copyright Act, I certify that I have obtained a written permission from the copyright owner(s) to include such material(s) in my thesis and have included copies of such copyright clearances to my appendix.

I declare that this is a true copy of my thesis, including any final revisions, as approved by my thesis committee and the Graduate Studies office, and that this thesis has not been submitted for a higher degree to any other University or Institution.

ABSTRACT

Bio-molecular networks have led to many discoveries in molecular biology. The most atypical of them are protein-protein interaction (PPI) networks. In PPI networks the nodes refer to proteins and edges refer to interactions between nodes. The comparison of PPI networks can be demonstrated as a powerful approach for examining interactions in these networks and predicting protein functions. This thesis contributes a new alignment algorithm for aligning three PPI networks. We examine how Three-Index Assignment Problem via Hungarian Pair Matching algorithm is used to maximize the complete match between the three networks to identify protein triplets with higher similarity. We have performed tests on PPI networks extracted from the IntAct database and IsoRank database. We experimentally show that the results obtained by our method have more biological significance in comparison to other methods and can be used in future to predict protein functions and complexes in PPI networks.

DEDICATION

To my parents and everyone who adds meaning to my life.

ACKNOWLEDGEMENTS

First and foremost I would like to thank my advisor Dr. Alioune Ngom, without whom this research would not have been possible. Dr Ngom's great research insight and caring personality makes him a perfect supervisor. I also would like to thank him for introducing me to the area of Bioinformatics and providing me with an initial idea of my research. His ideas, guidance, encouragement and patience not only made my research experience meaningful but also very enjoyable.

I would also like to thank Dr. Luis Rueda, Dr. Christie Ezeife and Dr. Kemal Tepe for kindly agreeing to be a part of my thesis committee. Also, I thank all my friends and colleagues who were always available for providing me with all the necessary support and helping me in completing my thesis successfully.

Finally, I am sincerely grateful to my parents who have been a pillar of support all through my life. Their belief that I could do a good research kept me motivated. Without their love and encouragement I would not have finished this thesis. This thesis is dedicated to them.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

1.1 Preface

Every living cell consists of proteins that continuously interact with each other to perform various functions. These cellular functions are not carried out by single proteins, but by proteins interacting with each other. Various techniques have been developed to understand these interactions. Due to the recent advances in the experimental biological techniques such as yeast-2-hybrid, tandem affinity purification and other high-throughput methods, a huge amount of protein-protein interaction (PPI) data is publicly available. The availability of large amount of data entails the researchers to devise new computational approaches to analyze these interactions and study the complex networks they form. The networks formed by these protein interactions are called protein-protein interaction (PPI) networks. The comparative analysis of PPI networks of various species can be very useful in the field of bioinformatics as it helps in revealing significant biological information. Unfortunately, unlike sequence comparison and alignment, comparing networks by aligning them is computationally hard and thus heuristic approaches must be devised. The purpose of this thesis is to provide new, better and efficient heuristic algorithm for aligning multiple networks

1.2 Background Study

In this section we discuss briefly the biological background and provide an introduction to proteins and protein-protein interaction networks.

1.2.1 Proteins

Proteins are large biological molecules consisting of one or more connected amino acid units. They are involved in practically every function performed by a cell. Some of the important examples of functional classes include: (1) enzymes, which catalyze, for example, the many of the reactions of metabolism; (2) structural proteins, such as collagen which is the main protein of connective tissue in animals; (3) regulatory proteins, such as transcription factors that regulate the transcription of genes; (4) signaling molecules, such as certain hormones, like insulin, and their receptors; and (5) defensive proteins such as antibodies of the immune system.

Recent advancements in high-throughput sequencing techniques, discovered the complete sequences of several genomes. However, the biological function of a large proportion of sequenced proteins remains to be identified. Moreover, a given protein may have more than one function, so many proteins that are known to be in some class may have as yet undiscovered functionalities. Predicting protein functions is one of the most important challenges of current computational biology research. To facilitate such research, various biological data could be used, including sequence, gene expression patterns, phylogenetic profiles, domain fusions and so on.

Proteins interact with each other to perform various functions. Hence, we see that protein-protein interactions operate at almost every level of cellular functions. Thus, knowledge bout protein functions can be inferred via protein-protein interaction studies. These implications are based on an idea that the function of unknown proteins can discovered by studying their interaction with a known protein target having a known

function. The study of protein interactions will help us understand how proteins function within the cell and predict protein functions of unknown proteins..

1.2.2 Protein-Protein Interactions

The Protein-Protein Interactions are referred to as the biochemical reactions between protein molecules. In all the living organisms the proteins interact with each other to perform various functions. For example, signal transduction within the cell takes place when chains of protein interactions occur many of which include kinase enzymes or proteins which react with other proteins to modify their function. Generally the proteins perform long-lasting interactions, creating protein complexes. A protein complex is a group of two or more proteins interacting with each other to perform a particular function. A single protein can be a part of various complexes. Various experimental methods have been described to identify the proteins participating in a complex to perform various functions.

There are proteins across various species that are similar to each other based on the shared ancestry. These proteins are referred to as orthologs. The orthology of the proteins in these species is detected via sequence similarity between their respective DNA or amino-acids sequences. The sequence similarity between these proteins is calculated using a sequence similarity alignment method. The most commonly method used for computing the sequence similarity is BLAST (Basic Local Alignment Search Tool), which aligns the sequences and computes a score called S-score of the alignment, and outputs the significance of the result as a number, the e-value (Expectation value). The e-value is defined as the number of different alignments with scores equivalent to or better

than the S-score that are expected to occur in a database. The lower the E-value, the more significant is the score. This score is used for determining the similarities between the proteins in large protein-protein interaction networks.

1.2.3 Protein-Protein Interaction Networks

The most commonly studied biological networks are known as Protein-Protein Interaction networks. (Figure 1.1 depicts an example of such a network). These networks are usually represented using undirected, weighted graphs where the nodes of these graphs represent the proteins and the edges represent the interactions between the proteins. The study of these networks becomes important to understand the various functions in a cell. As we know proteins never perform their function alone instead they interact with each other to perform various functions hence, studying and understanding these networks is one of the foremost challenges faced by the researchers today The study of the topology of the PPI networks gives us an insight about function of individual proteins in the networks as well as protein complexes.

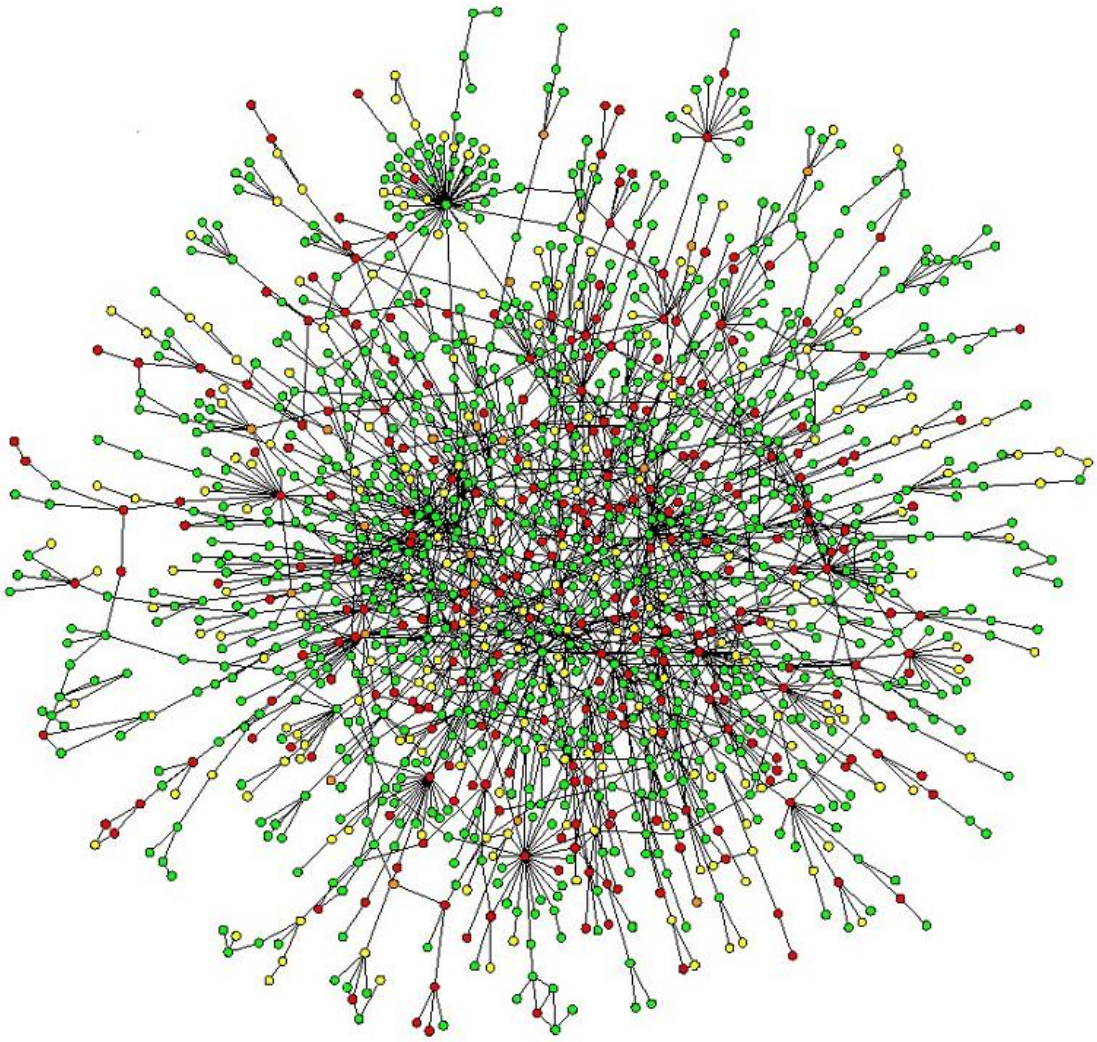


Figure 1.1: A map of protein-protein interactions in yeast (Barabàsi *et.al.*, 2004), which was based on early yeast two-hybrid measurements. A few highly connected nodes (which are also known as hubs) hold the network together

1.3 Thesis Outline

This thesis is organized as follows. The next chapter provides a very brief description of problem definition, current research motivation and previous approaches proposed for aligning protein interaction networks. Chapter 3 presents the explanation of Hungarian

algorithm which forms an important part of our method and also detailed description of a previous method PINALOG which serves as a basis for our new method. Chapter 4 describes our new algorithm for aligning three protein interaction networks using a solution to Three-Index Assignment Problem via Hungarian algorithm. Chapter 5 presents a description of datasets used and results of applying our method to PPI networks. It also presents the comparison of IsoRank's performance to our method. Finally, Chapter 6 gives a brief summary of the thesis and future work.

CHAPTER 2

PROBLEM DEFNITION AND PREVIOUS METHODS

2.1 Problem Definition and Network Alignment Problem

Proteins are considered as basic building blocks of all the cellular processes. Thus protein interactions perform all the activities that occur within the cell. Two or more proteins that are descendants of a common ancestral DNA sequence are known as homologs.Also, proteins in different species that evolve from a common ancestor are called orthologs. Thus, functions performed by proteins in different species may be related to each other. In order to identify similar related protein groups have attracted lot of researchers to compare PPI networks.

Let us consider a set of PPI networks of different species. In addition we also have protein sequence similarity data or function similarity data for every protein pair in the networks. The idea here is to find the sub-networks that are conserved across the species both in terms of proteins (similar sequence) and interactions (similar topology). The graphs are formally represented as :

$$G_i = (V_i, E_i) \quad \textbf{(2.1)}$$

$where\ 1 < i < k$ denote the PPI networks of species 1…..k, V_i is the set of proteins of species i and E_i is the set of protein-protein interactions.

Network alignment is the process of comparing `k networks, identifying regions of similarity and dissimilarity. The algorithms for network alignment can be divided into two categories.

- **Global Network Alignment -** The goal is to map every node in one network to a node in the other network. The mapping between the nodes of the two networks maximizes some kind of score. The score can be either sequence, function or topological similarity. The nodes that are not mapped to any node in the other network are present in the alignment without any matched partner. These types of alignment take into account the whole network into consideration and measure overall network conservation.
- **Local Network Alignment –** This alignment focus on finding the conserved sub-networks across the species, thus representing true functional modules. The goal is to find a local alignment that contains a sub-network from each species as well as the mapped nodes in the sub-networks. One disadvantage of aligning networks by this method is that the nodes aligned can overlap in different local alignments. The same node in one network might be aligned with different nodes in the other network. Also there is no way to know the overall similarity existing between the two networks.

The challenge in PPI network research is network comparison. Given two networks $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ the network alignment problem finds a mapping $f : V_1 \rightarrow V_2$ which matches similar nodes in the networks being compared. Aligning topologically similar nodes is called graph isomorphism. An isomorphism may not exist even if the two PPI networks are of the same size because of the biological variation in these biological networks. Thus network alignment problem includes sub-graph isomorphism problem.

Another way of comparing two networks is by forming a network alignment graph. The node of this graph is a collection of proteins, one from each network and the edges represent the conserved protein interactions between the networks (Figure 2.1).

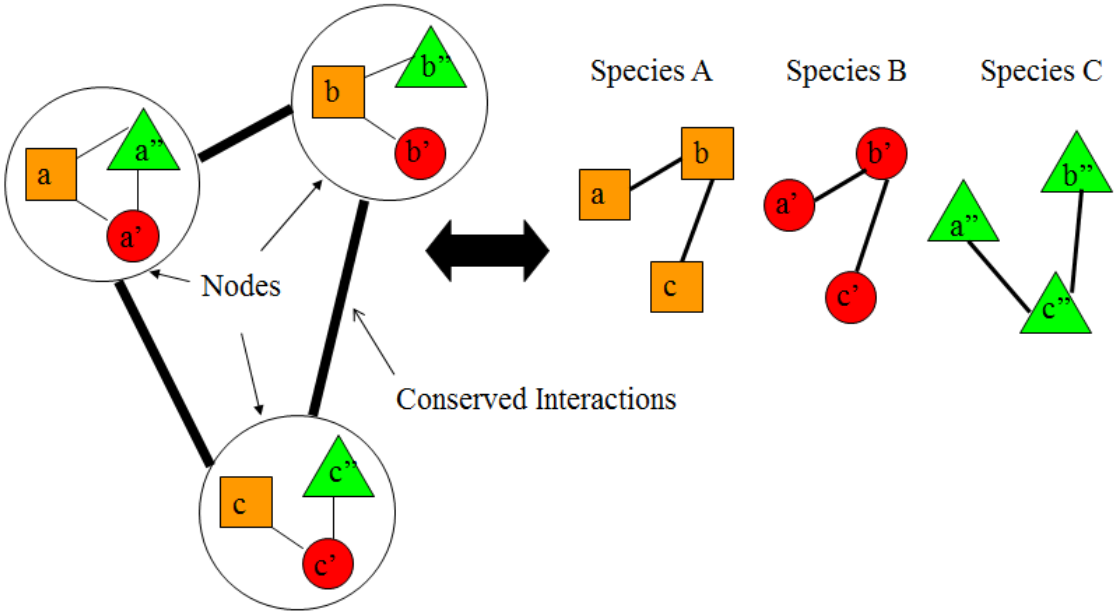


Figure 2.1: Example of Network Alignment Graph.

In the above example we see proteins of PPI networks of three species A, B and C. Let P_1 , P_2 and P_3 represent the sets of proteins in species A, B and C. For every homologous protein, $a \in P_1, a' \in P_2$ and $a'' \in P_3$, a node $P = (a, a', a'')$ is added to the network alignment graph.` The complexity of creating network alignment graph increases exponentially if more than two networks are being compared.

Hence, because of the large amount of PPI data available, we need an efficient method to match proteins of more than two the PPI networks.

2.2 Current Research Motivation

With the rapid advancement in technology the study of biological networks has become one of the primary focuses in the field of bioinformatics. The most commonly studied biological networks are the protein-protein interaction (PPI) networks. Since the proteins do not function alone they interact with one another to form protein complexes or functional module. The PPI networks can be graphically represented in the form of an undirected weighted graph denoted by $G = (V, E)$ where V is a set of nodes and $E = V \, x \, V$ is a set of edges. The nodes of the graph represent the proteins and the edges represent the interactions between the proteins. There are several crucial challenges faced in network alignment research.

Certain regions in PPI networks are expected to be conserved more than others during the course of evolution. The challenge faced by the researchers nowadays is to study and compare the given PPI networks in order to find the conserved sub-graphs. The conserved sub-graphs ensure that the proteins present in the two sub-graphs consist of protein that have similar functions and have similar interaction profiles. Since protein interaction networks are too large and complex it is essential devise an efficient alignment methods. The commonly used method is to generate a merged representation of the networks being compared, known as a network alignment graph. A network alignment graph consists of nodes that represent the set of proteins, one from each species and the edges represent the conserved protein-protein interactions across the species being compared. The alignment may be one-to-one correspondence or many-to-many correspondence between proteins. The network alignment method has been applied

by various authors successfully but its extension to more than two networks results in exponential growth of the alignment graph with the increase in number to species. The researchers are motivated to propose new algorithms to overcome this difficulty.

Another way of comparing two networks specified above, is the concept of graph isomorphism. Consider two networks $G_1 (V_1 , E_1)$ and $G_2 (V_2 , E_2)$, here the network alignment problem can be referred as to find a mapping function $f : V_1 \rightarrow V_2$ which aligns similar nodes on basis of topology. .But we know exact comparisons in biological networks is not possible because of the biological variations. Formulation of network alignment problem includes sub-graph isomorphism problem which is known to be NP-complete. Hence, network alignment problem is computationally hard and has to be addressed using heuristics.

We know that comparing networks can provide us with valuable insights to the biological information. Alignments can be used to transfer knowledge between protein networks such as predicting functions of unannotated proteins. This motivates us to formulate methods to align PPI networks in an efficient way in order to extract relevant biological information.

2.3 Thesis Contribution

The alignment of PPI networks helps in understanding the functioning of individual proteins. Many network alignment methods have been applied successfully by various authors for aligning two PPI networks. These methods align networks both globally and locally. The local network alignment (LNA) aims to identify small sub-networks that are

conserved across two species. In global network alignment (GNA), the goal is to associate proteins from two or more species in a global manner so as to maximize the overall conservation across the aligned networks.

Methods based on local network alignment methods include PathBLAST (Kelley et al., 2003), MaWISH (Koyutürk et al., 2006), which adopts the evolutionary models of match, mismatch and deletion of the proteins. The global alignment of networks proves to be more challenging due to the complexity and scale of the problem for example Graemlin 2.0 proposed by Flannick et al., 2009 formulates a model for protein duplication, deletion and mutation and aligns the network progressively using a hill-climbing algorithm, IsoRank by Singh et al., 2008 which aligns the networks by eigenvalue-based methods and PINALOG (Phan et. al. 2009) which is a pairwise alignment method that incorporates sequence, function and topological information to map the networks using Hungarian algorithm.

Analogous to global sequence alignment problem, in network alignment problem we aim to find the overall best match between the PPI networks using network topology, sequence similarity and function similarity between proteins of the networks. In this thesis propose a method for aligning multiple species. The method is based on PINALOG; a global pairwise network alignment method which is extended to perform multiple network alignment. We introduce a method which is capable of aligning three protein interaction networks based on combination of sequence similarity and function similarity between the proteins of the networks and later incorporating network topology. The pairwise alignment is extended to perform multiple alignment by using a solution to

Three-Index assignment problem via Hungarian algorithm and thus obtain overall best match between three networks. This thesis provides a flexible and scalable(in terms of computational running time) method for comparing and aligning protein interaction networks.

2.4 Previous Methods for Aligning PPI Networks

A variety of methods have been proposed for PPI network alignment. The network alignment method has been successfully implemented by various authors for pairwise alignment of networks. However, aligning more than two networks has proven to be difficult because of the exponential growth of the alignment graph with the number of species. Thus alignment of multiple networks is a challenge faced by the researchers today.

2.4.1 Pairwise Network Alignment Methods

2.4.1.1 *PathBlast*

One of the first successful algorithms for pairwise local network alignment is Path-BLAST . This method searches for high-scoring alignments of pathways from two networks as shown in Figure 2.2. It pairs proteins along a pathway from one network with their *homologues*, i.e., proteins that are descendants of common ancestry, from another network. This algorithm can be described as follows. First, the "global network alignment graph" between two networks is constructed as illustrated in Figure 2.2b. As discussed above each vertex of this graph represents a pair of proteins from two networks with similar protein sequences (BLAST E-value $\leq 10-2$). An edge between nodes (A, a)

and (*B, b*) in this "global network alignment graph" can be of the three types: (i) "direct" - both edges (*A,B*) and (*a, b*) are present in the input PPI networks, (ii) "gap" – only one of the edges (*A,B*) or (*a, b*) is present in the data and (iii) "mismatch" - (*A,B*) and (*a, b*) are absent in both networks. Similar to to sequence alignment method, this algorithm also allows for gaps and mismatches in the alignments. Then, for each path *P* in the "global alignment graph," its log-likelihood score is defined as

$$S(P) = \sum_{v \in V(P)} \log \frac{p(v)}{p_{random}} + \sum_{e \in E(P)} \log \frac{q(e)}{q_{random}} \qquad \textbf{(2.2)}$$

where $p(v)$ is the probability that the proteins in the pair corresponding to v are true homologues, given their pairwise sequence similarity measured as BLAST E-value, and $q(e)$ is the probability that the protein-protein interaction represented by e is real. The $q(e)$ value is estimated based on the number of studies that confirmed interaction e and the quality of the experiments that confirmed it. p_{random} and q_{random} are expected values of $p(v)$ and $q(e)$ taken over all nodes and edges in the "global alignment graph," respectively. Based on this scoring function, a dynamic programming algorithm is used to find high-scoring pathway alignments of size L in the global alignment graph.

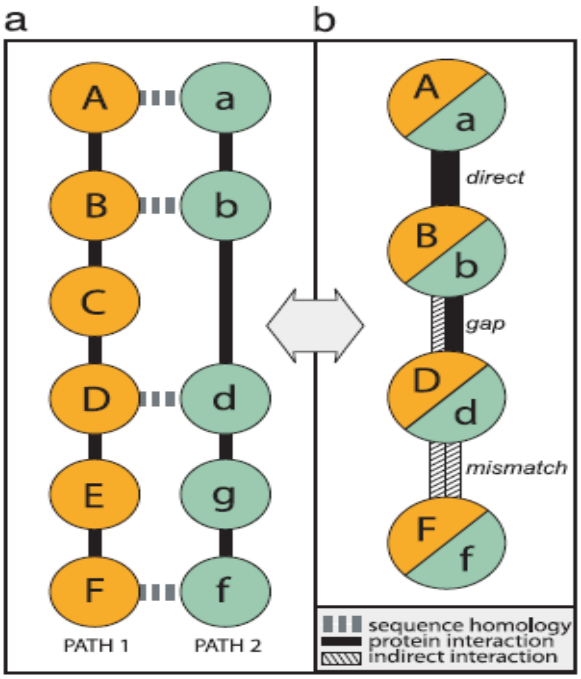


Figure 2.2: (a) An example of pathway alignment. Capital letters represent nodes from one network and small letters represent nodes from another network. Dotted horizontal lines represent local alignments that link proteins with high sequence similarity. Gaps (e.g. at node "C") and mismatches (e.g., at nodes "E" and "g") are allowed in the alignment. (b) The two paths from panel (a) combined into the global alignment graph. This figure is taken from (Kelly *et.al.*, 2003).

PathBLAST was used to identify orthologous pathways between yeast *S. cere- visiae* and bacteria *H. pylori*.. Later, Suthram *et al.* (22) used PathBLAST to compare the PPI networks of *Plasmodium falciparum* (the pathogen responsible for over 90% of human deaths from malaria) with PPI networks of model eukaryotic organisms: the budding yeast *Saccharomyces cerevisiae*, the nematode worm *Caenorhabditis elegans*, the fruitfly *Drosophila melanogaster* and the bacterial pathogen *Helicobacter pylori*. Based on their alignments of these networks using PathBLAST, they found 29 highly connected protein complexes specific to the network of the pathogen. However, only 3 of them were conserved in the yeast. Since yeast, fly and worm share a substantial amount of conserved complexes with each other (as Suthram *et al.* (22) revealed using PathBLAST), this suggests that the PPI network of this pathogen encodes significant functional differences worth of further investigation.

2.4.1.2 MaWISh

Another method for pairwise local alignment of PPI networks is MaWISh (Maximum Weight Induced Subgraph). This algorithm is based on the duplication/ divergence models. It is based on understanding the evolution of protein interactions. Analogous to sequence alignment method, the concept of match, a gap and a duplication event are defined, as well as the corresponding scores for these events. Firstly, the "global alignment graph" is constructed from the PPI networks being aligned. This "global network alignment graph" is conceptually similar to those used by Path-BLAST. Its node set consists of all pairs of nodes $v = (v_1, v_2) : v_1 \in V(G_1), v_2 \in V(G_2)$ such that $S(v_1, v_2) > 0$, where *S* defines the likelihood that (v_1, v_2) are orthologs and is defined as

$$S(v_1, v_2) = P(E(v_1, v_2) \leq \tilde{E} | \mathbb{O}) = \frac{|\, v_1, v_2 \in \mathbb{O} : E(v_1, v_2) \leq \widetilde{E}|}{|\mathbb{O}|} \quad \textbf{(2.3)}$$

where $E(v1, v2)$ is a BLAST E-score for protein sequences of (v_1, v_2), E is manually chosen threshold and $\mathbb{O}$ is a set of all known orthologous pairs of nodes from two networks. The edges of their "global alignment graph" are weighted, with weights equal to

$$w(uv) = \mu(u_1u_2, v_1v_2) + \nu(u_1u_2, v_1v_2) + \delta(u_1, v_1) + \delta(u_2, v_2) \quad \textbf{(2.4)}$$

where $\mu, \nu \; and \; \delta$ are scores for match, mismatch and duplication events, respectively. The goal of MaWISh algorithm is to find an induced subgraph of maximum weight in the "global alignment graph.".

Koyuturk *et al.* 2005 used MaWISh to perform pairwise alignments of yeast (*S.cerevisiae*), worm (*C. elegans*) and fruitfly (*D. melanogaster*) PPI networks. Aligning yeast and fly PPI networks by using MaWISh, they identified 412 conserved

subnetworks. Note that these alignments are very "local" in the sense that these conserved subnetworks contain about 10 nodes each.

2.4.2 Multiple Network Alignment Methods

2.4.2.1 *Graemlin*

Graemlin 2.0 is a global network alignment algorithm for multiple network alignments. This algorithm performs both global and local network alignments. It obtains its parameters of scoring function from the data and its complexity scales linearly with the number of networks in the multiple network alignment. Flannick *et al.* 2009 define a multiple network alignment as an equivalence relation a over the nodes of $V = V_1 \cup V_2 \cup \ldots \cup V_n$. Example of such equivalence relation for four networks is given in Figure 2.3. It is a transitive relation and it partitions V into disjoint equivalence classes of orthologous proteins. The global alignment is an equivalence relation over all nodes in V, whereas the local alignment is a relation over a subset of nodes in V.

The scoring function used by Graemlin 2 computes the features of the global network alignment to a numerical feature vector of the form

$$f(a) = \begin{bmatrix} \sum_{[x] \in a} f^N([x]) \\ \sum_{[x],[y] \in a \;\; [x] \neq [y]} f^E([x] \;,\; [y]) \end{bmatrix} \quad \textbf{(2.5)}$$

where $[x]$ represents an equivalence class of nodes under alignment a, and f^N and f^E are node and edge feature functions scoring several evolutionary events. The score of alignment a is then given by $s(a) = wf(a)$ where w is a parameter vector to be learned. The pairwise node feature function computes and scores the following evolutionary events:

- *Protein present* = which denotes the existence of a protein in both species.
- *Protein count* = the count of proteins that exist in both species.
- *Protein deletion* = a loss of a protein in one of the two species.
- *Protein duplication* = *the* duplication of a protein in one of the two species.
- *Protein mutation* = *the* divergence in sequence of two proteins in different species.
- *Paralog mutation* = the divergence in sequence of two proteins in the same species.

For edge feature function two evolutionary events are considered:

- *Edge deletion*= a loss of an interaction between two pairs of proteins in different Species
- *Paralog edge deletion* = a loss of an interaction between two pairs of proteins in the same species.

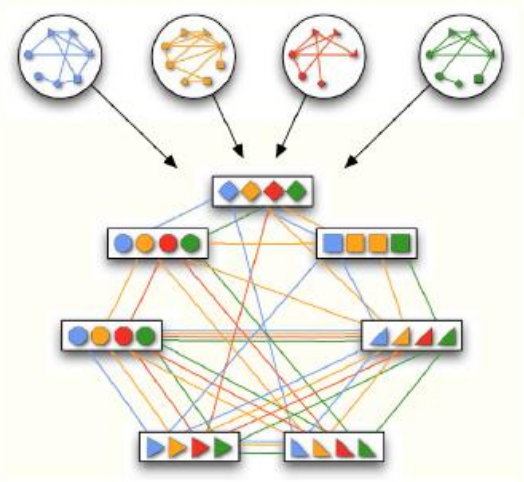


Figure 2.3: A graph representation of the equivalence relation corresponding to the multiple alignment of four PPI networks.

All these evolutionary events for nodes and links in the network are defined for a pair of networks. Hence, to efficiently generalize these scores for multiple network alignment,

Graemlin uses phylogenetic trees of species being aligned and incorporates evolutionary distance between species into its scoring function. Parameter *w* is learned from the example set of networks with known optimal alignments. Once the optimal vector of parameters has been learned, Graemlin uses iterative hill climbing technique to find the optimal (the one with the highest score) global alignment.

To test the performance of Graemlin, its authors performed several pairwise alignments of yeast, human, mouse and different bacteria PPI networks. They also performed a three-way alignment of yeast, worm and fly networks from DIP (34) as well as six-way alignment of *E. coli*, *S. typhimurium*, *Vibrio cholerae*, *Campylobacter jejuni*, *Helicobacter pylori*, and *C. crescentus* PPI networks. To measure the sensitivity and specificity of their algorithm, Flannick *et al.* compared the alignments produced by Graemlin with KEGG Orthology (KO) groups. They also evaluated Graemlin and all previously discussed algorithms on the same datasets and showed that Graemlin is both more sensitive and more specific than all of the algorithms discussed earlier in this chapter. In order to perform global alignment, Graemlin requires a lot of information as input: (i) node sequence similarity scores that estimates evolutionary events, (ii) the phylogenetic tree of species being aligned for multiple network alignment, and (iii) a training set consisting of several networks and "correct" alignments between them to learn the parameter values.

2.4.2.2 *IsoRank*

One of the most advanced algorithms for global network alignment up to date is IsoRank (Singh et. al., 2006). It is inspired by Google's PageRank method and is based on the fact that two nodes should be aligned together only if their neighbors can also be well matched together. Its method is formalized using the eigenvalues problem.IsoRank is the first *global* network alignment algorithm.

Given two networks G_1 and G_2, the algorithm works in two stages: (i) $\forall i \in V(G_1)$ and $\forall j \in V(G_2)$ it computes the scores R_{ij} of matching node *i* with node *j*, (ii) it constructs a global network alignment by extracting from vector *R*, high-scoring pairwise mutually-consistent matches. Given $n_1 \times n_2$ dimensional scores vector *R* is subject to the following constraints

$$R_{ij} = \sum_{u \in N(i)} \sum_{v \in N(j)} \frac{R_{u,v}}{|N(u)||N(v)|}, \ \forall i \in V(G_1), \ j \in V(G_2) \qquad \textbf{(2.6)}$$

where $N(u)$ is a neighborhood of node *u*. This equation can be written in the matrix form.

$$R = AR \qquad \textbf{(2.7)}$$

where $A[i, j][u, v] = 1/|N(u)||N(v)|$ if $(i, u) \in E1$ and $(j, v) \in E2$, and $A[i, j][u, v] = 0$ otherwise. Note that *A* is a stochastic matrix (i.e., each of its columns sum to 1), so its principle eigenvalue is 1. The matrix *A* is of size $n_1 n_2 \times n_1 n_2$, and is very sparse and *R* can be efficiently computed using some iterative technique such as the power method. The above equations is modified to include pairwise information about node similarity (i.e., sequence information) as shown in Equation 2.8

$$R = \alpha AR + (1 - \alpha)E \qquad \textbf{(2.8)}$$

where E is a matrix with pairwise sequence scores between the nodes and α is a user defined parameter which controls the contribution of sequence versus topology information in the alignment. After computing the value of R the global network alignment is then constructed by interpreting R as a weighted bipartite graph and finding the maximum-weight bipartite matching. IsoRank constructs global alignment between yeast *S. cerevisiae* and fly *D. melanogaster* PPI networks. The common subgraph, as revealed by this alignment, consists of 1,420 edges present in both species. The authors use their alignment to identify functional orthologs between yeast and fly.

For multiple alignment, first stage of the algorithm remains the same, but is executed for all pairs of networks creating a *k*-partite graph. Thus, the second stage was changed to find the optimal solutions of the *k*-partite matching. This version of IsoRank was used to perform the alignment of the five PPI networks, of yeast, fly, mouse, worm and human. The common subgraph constructed by this alignment had 1,663 edges that were supported by edges in at least two (out of five) aligned PPI networks, and only 157 edges that were supported by at least three PPI networks (i.e., species). Based on this alignment, functional orthologs predictions were made.

CHAPTER 3

RELATED WORK

3.1 Preface

In this chapter we discuss the details of the methods applied to find out the mapping between the protein interaction networks. We begin by describing an algorithm called Hungarian algorithm. As discussed earlier the challenge faced in aligning protein interaction networks is to find an optimal alignment algorithm which is fast and accurate. In the network alignment of two or more networks we focus on identifying regions of similarity and dissimilarity. Since in our thesis we focus on global alignment we need to find a mapping that maximizes the total network score. The score can be sequence similarity, or functional similarity between the proteins or can be based on the topology of the networks. This mapping can be achieved by considering this problem as a maximum weight matching problem. Finding maximum weight matching is called an assignment problem which is one of the most fundamental optimization problems. A very famous assignment problem was developed by Kuhn (1955) which maximizes/minimizes the total cost called Hungarian Algorithm. We then discuss a pairwise protein interaction network alignment method known as PINALOG that provides a basis for our approach.

3.2 Hungarian Algorithm

The standard assignment problem is referred to as the problem to find a one-to-one matching between n tasks and n agents, in order to optimize the total cost of the assignments. The objective is either to maximize or minimize the total cost. In this thesis

we wish to find an optimal assignment which maximizes the total cost function. The classical example of assignment problems is assigning jobs to workers. Hungarian method is the most popular method which solves the assignment problem in polynomial time. It was developed and published by Harold Kuhn in 1955. Consider an assignment problem in which we want to assign N tasks to N agents where each agent is assigned to at the most one task. The objective function is to maximize the total cost of assignments. The mathematical model for the assignment problem may be given as:

$$\boldsymbol{max} \sum_{\boldsymbol{i=1}}^{\boldsymbol{N}} \sum_{\boldsymbol{j=1}}^{\boldsymbol{N}} \boldsymbol{c_{ij} x_{ij}} \qquad \textbf{(3.1)}$$

Subject to: $\sum_{j=1}^{N} x_{ij} = 1 \qquad \forall\, i \in N$

$$\sum_{i=1}^{N} x_{ij} = 1 \qquad \forall\, j \in N$$

$$x_{ij} = 0 \; or \; 1$$

Where $\{c_{ij}\}_{N\,x\,N}$ is the cost of assigning agent $\boldsymbol{i}$ to task $\boldsymbol{j}$ and $\{x_{ij}\}_{N\,x\,N}$ is the resulting binary matrix, where $\boldsymbol{x_{ij}}$ = 1 if and only if an agent $\boldsymbol{i}$ is assigned to task $\boldsymbol{j}$.

In terms of graph theory we can represent this problem as a maximum weight bipartite matching. A bipartite graph $G = (U, V, E)$ is a graph whose vertices can be divided into two disjoint sets U and V such that each edge $(u_i, v_j) \in$ E connects a vertex $u_i \in U$ and $v_j \in V$. The network representation in the form of a bipartite graph is given below.

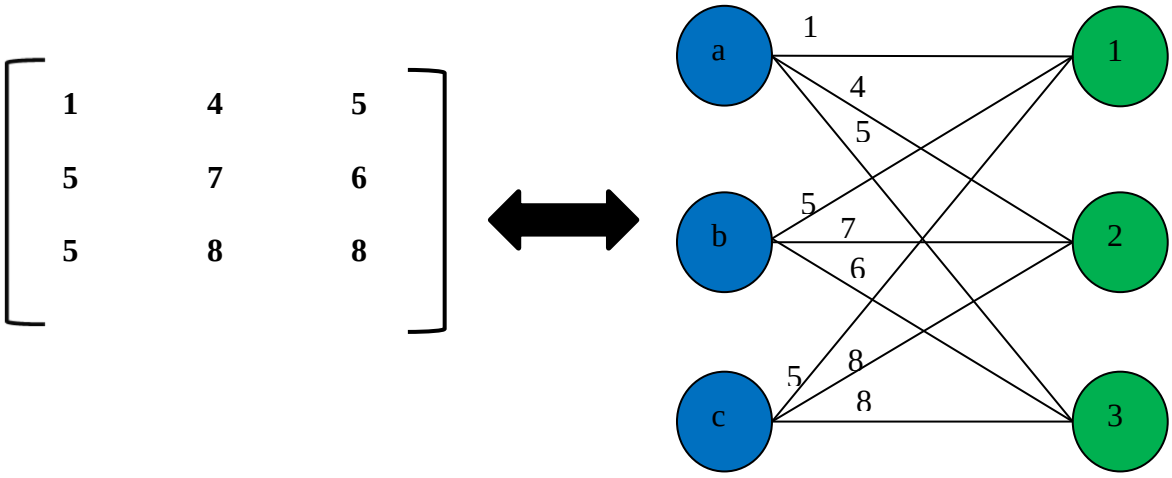


Figure 3.1 Matrix representation of a complete weighted bipartite graph.

In terms of protein interaction networks the weight on the edges of the graph can be a sequence similarity or a functional similarity score between the proteins. Hence we can create a matrix using the similarity score and find a matching between two protein networks. The maximization and minimization problems are essentially the same, however one can be transformed into the other by replacing the weight on each edge with an inverse of the weight.

3.2.1 Preliminary

Given a weighted complete bipartite graph $G = (X, Y, E)$ where $|X| = |Y| = n$, $E = (X\ x\ Y)$ and edge (x, y) has weight $w(x, y)$ we want to find a matching M from X to Y with the maximum weight.

Before we proceed further we will discuss some theoretical ideas used in the algorithm.

- We assume that all the weights are non-negative

$$w(x,y) \geq 0, \quad \forall x \in X \; and \; \forall y \in Y \tag{3.2}$$

- **Vertex Labeling:** It is defined as a function $l : V \rightarrow R$ that assigns a number called label to each vertex in the graph. A label is called feasible if it satisfies the following condition

$$l(x) + l(y) \geq w(x,y), \; \forall x \in X, \forall y \in Y \tag{3.3}$$

- **Vertex and Set neighborhood:** Consider a vertex $v \in V$, then all the vertices that share an edge with the vertex v (neighborhood) can be given by the equation 3.4

$$J_G(v) = \{ u \mid (v,u) \in E \} \tag{3.4}$$

Let $S \subseteq V$. Then all the vertices that share an edge with the vertex in S (neighborhood) is given by the equation 3.5.

$$J_G(S) = \bigcup_{v \in S} J_G(v) \tag{3.5}$$

- **Equality Graph:** A given graph $G_l = (V, E_l)$ where G_l is a sub-graph of G is called an equality graph if it consists of only those edges from the bipartite matching which allow the edges to be perfectly feasible. Thus equality includes only those edges that satisfy the flowing equation

$$(x,y) \in E_l \iff (x,y) \in E \wedge l(x) + l(y) = w(x,y) \tag{3.6}$$

- **Alternating path and alternating tree:** Consider a matching $M \; where \; (M \subseteq E)$. A matching can have matched vertices and

unmatched (free/exposed) vertices. A matched vertex $v \in V$ can be called matched if it satisfies the equation otherwise it is called exposed.

$$\exists x \in X : (x, v) \in M \vee \exists y \in Y : (v, y) \in M \quad \textbf{(3.7)}$$

Thus a path P is called an alternating path if its edges alternate between $M \; and \; E \backslash M$.It begins at a free vertex and alternate between free and matched edges.

An alternating tree is defined as a tree whose root vertex is a free vertex and every path that starts from that root is alternating.

- **Augmenting Path:** A path is said to be augmenting if it is an alternating path starting and ending at a free vertex.

3.2.2 The Algorithm

Below we describe the algorithm to find a maximum matching in the given bipartite graph.

Step 1: We start with assigning a feasible vertex label to all the vertices in the graph and determine the equality sub-graph G_l. The initial labelling is calculated by the equation

$$l(x) = \max(w(x, y), (x, y) \in E \quad \textbf{(3.8)}$$

$$l(y) = 0, y \in Y$$

Step 2: Check if M is perfect then stop as we have our optimal solution. Otherwise, for some exposed $x \in X$ we set $S = \{ x \}$ and $T = \{ \}$. Here x is considered as the root of the alternating tree that we are going to build.

Step 3: If $J_G\ (S)\ \neq\ T$ then go to Step 4. Otherwise if $J_G\ (S)\ =\ T$ then calculate Δ by equation

$$\Delta = \min(\ l(x) + \ l(y)\ -\ w(x,y)\ ,\ x\ \in S\ and\ y\ \in\ Y \backslash \in T \tag{3.9}$$

After calculating the Δ update the existing labels according to the equation

$$l'(v)\ =\ \begin{cases} l(v) - \ \Delta\ ,\ v\ \in S \\ l(v)\ +\ \Delta,\ v\ \in T \\ l(v),\quad otherwise \end{cases} \tag{3.10}$$

After calculating the updated labels replace the equality graph G_l with $G_{l'}$.

Step 4: In this step we choose a vertex $y\ \in\ T \backslash J_G(S)$. If y is matched in M with some vertex say z add the edge (z, y) to the alternating tree and update S and T by following equations and go to Step 3.

$$S\ =\ S \cup \{z\} \tag{3.11}$$

$$T\ =\ T \cup \{y\}$$

Otherwise if y is exposed, there will be an alternating path from x to y and we use this path and a larger matching M' in G_l. We replace M by M' and go to Step 2.

3.2.3 Runtime Analysis:

The time complexity of Hungarian algorithm is $O(n^3)$.

The size of the matching *M* never decreases. At each iteration we either increase the size of *T*, or we update the labels, which will cause us to increase the size of *T* in the next iteration. So after 2*n* iterations, the size of *T* will be *n*. Since *T* cannot grow anymore, we will have to increase the size of *M*. But the size of *M* is at most *n*, so the algorithm will

finish after at most $O(n^2)$ iteration. An iteration can be executed in time $O(n)$, so the total running time is bounded by $O(n^3)$.

3.2.4 A Walk through Algorithm

Consider a $4\,x\,4$ weighted bipartite graph. The figure shows the weight matrix for the given graph

9	2	8	1
2	5	2	6
2	1	5	3
1	1	1	1

Figure 3.2: Example of a weight matrix

Step 1: First we do vertex labeling and find the maximum match M using equality graph condition.

	y_1	y_2	y_3	y_4	$l(y_i)$
x_1	9	2	8	1	9
x_2	2	5	2	6	6
x_3	2	1	5	3	5
x_4	6	1	1	1	6
$l(x_i)$	0	0	0	0	

Figure 3.3: Example of a weight matrix with vertex labels.

After updating the labels we find the maximum matching in the matrix. From the matrix we get $M = \{(x_1, y_1), (x_2, y_4), (x_3, y_3)\}$. In the following we show the edges that are matched. The matched edges are shaded and the edges that have not been matched are dashed.

	y_1	y_2	y_3	y_4	$l(y_i)$
x_1	9	2	8	1	9
x_2	2	5	2	6	6
x_3	2	1	5	3	5
x_4	6	1	1	1	6
$l(x_i)$	0	0	0	0	

Figure 3.4: Equality graph of the given example.

Step 2: From the above figure we see that x_4 is not matched. Thus according to Step 2 of the algorithm we set $S = \{x_4\}$ and $T = \{\}$.

Step 3: Here we compute $J_G(S)$. We assign $J_G(S) \neq \{y_1\}$. Since $J_G(S) \neq T$,go to Step 4.

Step 4: Now we choose $y = y_1$ in $J_G(S) - T$. We see that the vertex y_1 is matched with x_1 in the matching M. Thus we add x_1 to the set S and the vertex y_1 to T. After updating the values we get $S = \{x_4, x_1\}$ and $T = \{y_1\}$.Go to Step 3 again.

Step 3: We compute $J_G(S) = \{y_1\}$. Since $J_G(S) = T$, we compute Δ.

$$\Delta = \min(l(x_1) + l(y_3) - w(x_1, y_3) = 1$$

After calculating the value for Δ we decrement the labels for vertices x_1 and x_4 according to the equation by 1 and on the other hand we increment the label of vertex y_1 by 1. The figure below shows the updated labels.

	y_1	y_2	y_3	y_4	$l(y_i)$
x_1	9	2	8	1	8*
x_2	2	5	2	6	6
x_3	2	1	5	3	5
x_4	6	1	1	1	5*
$l(x_i)$	1*	0	0	0	

Figure 3.5: Updated Labels.

We compute $J_G\,(S) = \{\, y_1,\ y_3\}$.

Step 4: We choose $J_G\,(S) = \{\, y_3\,,\ y_1\}$ in $J_G\,(S) - T$**.** We again see that vertex y_3 is matched with x_3 in the matching M. Thus we add x_3 to the set S and the vertex y_3 to T. After updating the values we get $S\ =\ \{\, x_4\,,\ x_1\,,\ x_3\}\ and\ T =\ \{y_1\,, y_3\,\}$. Go to Step 3 again.

Step 3: We compute $J_G\,(S) = \{\, y_1\,, y_3\}$. Since $J_G\,(S) =\ T$, we compute Δ.

$$\Delta =\ \min(\, l(x_3) + l(y_4)\ -\ \ w(x_3, y_4)\ =\ 2$$

After calculating the value for Δ we decrement the labels for vertices $x_1\,, x_3\ and\ x_4$ according to the equation by 2 and on the other hand we increment the label of vertex $y_1\ and\ y_3$ by 2. The figure below shows the updated labels.

	y_1	y_2	y_3	y_4	$l(y_i)$
x_1	9	2	8	1	6*
x_2	2	5	2	6	6
x_3	2	1	5	3	3*
x_4	6	1	1	1	3*
$l(x_i)$	3*	0	2*	0	

Figure 3.6: Updated Labels.

We compute $J_G(S) = \{y_1, y_3, y_4\}$

Step 4: Now we choose $y = y_4$ in $J_G(S) - T$. We see that the vertex y_4 is matched with x_2 in the matching M. Thus we add x_2 to the set S and the vertex y_4 to T. After updating the values we get $S = \{x_4, x_1, x_2, x_3\}$ and $T = \{y_1, y_3, y_4\}$.Go to Step 3 again.

Step 3: We compute $J_G(S) = \{y_1, y_3, y_4\}$. Since $J_G(S) = T$, we compute Δ.

$$\Delta = \min(l(x_2) + l(y_2) - w(x_2, y_2) = 1$$

After calculating the value for Δ we decrement the labels for vertices x_1, x_2, x_3 and x_4 according to the equation by 1 and on the other hand we increment the label of vertex y_1, y_3 and y_4 by 1. The figure below shows the updated labels.

	y_1	y_2	y_3	y_4	$l(y_i)$
x_1	9	2	8	1	5*
x_2	2	5	2	6	5*
x_3	2	1	5	3	2*
x_4	6	1	1	1	2*
$l(x_i)$	4*	0	3*	1	

Figure 3.7: Updated Labels.

Step 4: Now we choose $y = y_2$ in $J_G(S) - T$. We see that the vertex y_2 is not matched in the matching M. Thus there exists an augmenting path from x_4 to y_2. The following figure shows the tree which has x_4 as its root.

‘

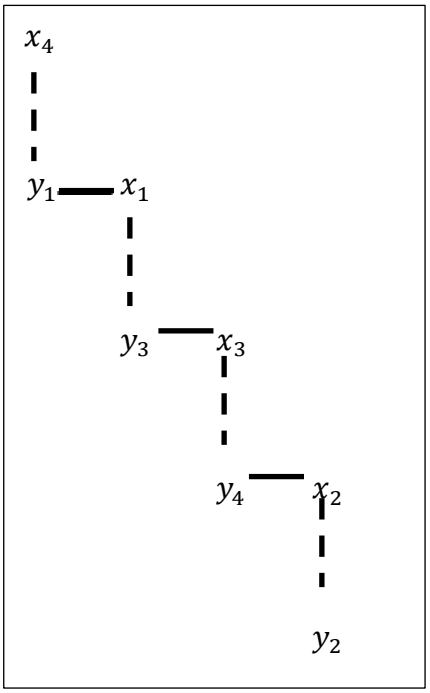


Figure 3.8: Alternating Tree. The vertical lines show the non-matched edges whereas the horizontal line show matched edged.

Thus the alternating path is

$P = \{(x_4, y_1), (x_1, y_1), (x_1, y_3), (x_3, y_3), (x_3, y_4), (x_2, y_4), (x_2, y_2)\}.$

Here we construct a new matching M' by equation

$$M' = (M \cup P) - (M \cap P)$$

$$= \{(x_4, y_1), (x_1, y_3), (x_3, y_4), (x_2, y_2)\}$$

We set $M = M'$ and go to Step 2.

	y_1	y_2	y_3	y_4	$l(y_i)$
x_1	9	2	8	1	5*
x_2	2	5	2	6	5*
x_3	2	1	5	3	2*
x_4	6	1	1	1	2*
$l(x_i)$	4*	0	3*	1	

Figure 3.9: Final Assignment

Step 2:

We see that M is perfect and we get maximum weighted matching with total weight = 22, hence we stop the algorithm.

3.3 PINALOG

Several methods have been described for aligning two protein interaction networks. One of the recent methods proposed by Phan *et. al., (*2012) for aligning networks of two species is called PINALOG. It is a global alignment method that takes into account both the protein sequence as well as the functional similarity between the proteins of two species. The sequence similarity between the two proteins is calculated using the Blast bit score whereas the functional similarity is calculated using the Gene Ontology annotations.

The following section will explain in detail the approach followed by PINALOG method to align two protein interaction networks. Before describing let us assume A and B are two protein-protein interaction networks of two species. The proteins in both the networks are represented using the notation a_i and b_j where a_i is the i^{th} protein in network A and b_j is the j^{th} protein in network B. The sequence similarity of two proteins a_i and b_j is given by the equation

$$\boldsymbol{s_{seq}}(a_i, b_j) = \frac{\boldsymbol{S}(\boldsymbol{a_i}, \boldsymbol{b_j})}{\sqrt{\boldsymbol{S}(\boldsymbol{a_i}, \boldsymbol{a_i})\, \boldsymbol{S}(\boldsymbol{b_j}, \boldsymbol{b_j})}} \qquad \textbf{(3.12)}$$

$S(a_i, b_j)$ is the BLAST bit score value when aligning a_i and b_j. The functional similarity $\boldsymbol{s_{func}}(a_i, b_j)$ of two proteins is calculated by the method proposed by Schlicker *et al,*. (2006). The detailed description of calculating this score is discussed in next chapter.

3.3.1 Methodology

The algorithm for aligning two networks *A* and *B* is divided into three steps.

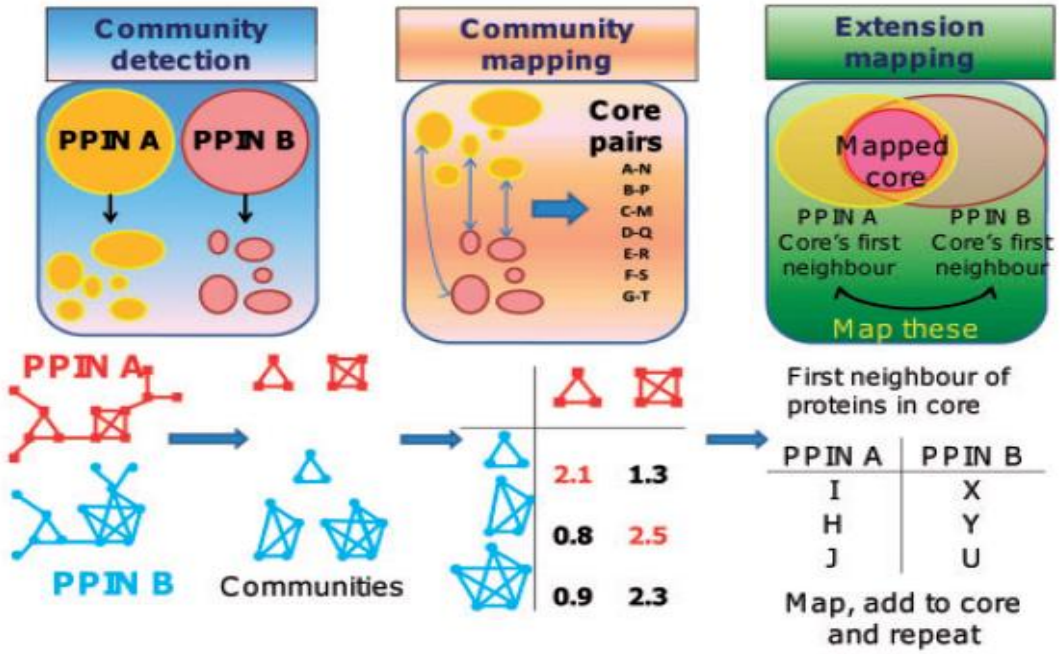


Figure 3.10: (i) Community Detection (ii) Community Mapping (iii) Extension Mapping

3.3.1.1 Community Detection

In the first step the algorithm focuses on finding the highly connected sub-networks within the input networks. The assumption is rather than aligning the whole PPI network it is efficient and reliable to align two protein interaction networks by first finding highly similar protein pairs extracted from the highly connected sub-networks.

In biological networks these highly connected sub-networks are referred to as communities. Thus a community is a sub graph of a network where a set of nodes are densely connected with each other in comparison with the rest of the network. An example of communities in a network is shown below (Fortunato *et. al*., 2010).

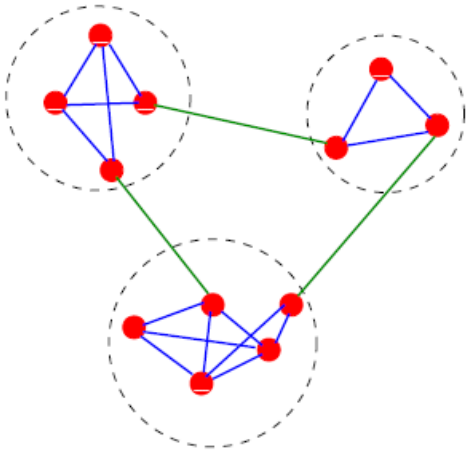


Figure 3.11: Example showing three communities in a network.

The process of finding communities from protein interaction networks is called clustering. Clustering of PPI networks is the task of grouping a set of proteins into groups (clusters/communities) so that the proteins in the same community are similar to each other than those in other communities. Several methods have been proposed to detect the communities. PINALOG uses CFinder (Palla *et. al*. 2005) which detects overlapping communities in the given networks. This method of clustering is based on Clique Percolation method and constructs communities by merging adjacent cliques. The detailed description of CFinder is given in section 4.2.1.

3.3.1.2 Community Mapping

After the communities have been detected using CFinder this step maps the communities having the highest similarity score. The communities of two networks $A\ and\ B$ are mapped using Hungarian algorithm. In order to obtain the optimal match between the communities, a community similarity matrix is formulated. The values $F(\ C_i^A\ ,\ C_j^B\ )$ of

this matrix are the sum of similarities between the proteins pairs obtained during optimal mapping (OptMap) of proteins in community C_i^A in species A with proteins in community C_j^B in species B using Hungarian algorithm. Thus the score of community similarity matrix computed using Hungarian algorithm is given by the equation 3.13

$$F\left(C_i^A, C_j^B\right) = \sum_{\substack{a_k \in C_i^A, b_l \in C_j^B \\ (a_k, b_l) \in OptMap}} s(a_k, b_l) \quad \textbf{(3.13)}$$

The matrix constructed using theses scores is then used to obtain optimal assignment of communities in both the networks and the maximized community scoring function $F(core)$:

$$F(core) = \sum_{C_i^A \subset A, C_j^B \subset B} F\left(C_i^A, C_j^B\right) \quad \textbf{(3.14)}$$

$F(core)$ is the total similarity score obtained after matching the communities using Hungarian algorithm. After obtaining the matched communities, protein pairs matched in these communities are extracted. These matched proteins are referred to as core proteins. A filtering step is performed and only 15% of these core pairs are retained and the rest are discarded.

3.3.1.3: Extension Mapping

Extension mapping step includes the topology of networks in the alignment. The neighbors of the core proteins extracted above are considered as candidates for this step for adding to the alignment. In addition to protein sequence and functional similarity, topological similarity in the protein interaction networks is also included in the form of neighborhood similarity. The set of all first neighbors (proteins separated by one interaction) and second neighbors (proteins separated by two interactions) of a_i in A and

b_j in B are denoted by $N(a_i)$ and $N(b_j)$. Let $d(a_k, a_i)$ denote the distance between a_k *and* a_i in a network. The similarity between a_i and b_j in extension mapping is then defined as

$$s_{ext}(a_i, b_j) =$$

$$s(a_i, b_j) + \sum_{\substack{a_k \in N(a_i) \\ b_l \in N(b_j) \\ (a_k,\ b_l) \in core}} \frac{1}{(d(a_k, a_i) + 1)(d(b_l, b_j) + 1))}\ s(a_k,\ b_l) \quad \textbf{(3.15)}$$

This step aims at adding more protein pair neighbors to the alignment. The optimal equivalence is obtained by using Hungarian method. These candidates are then added to the core and this process is repeated until no more pairs can be added.

PINALOG aligns different pairs of protein interaction networks from human , yeast, fly, worm and mouse and compare its results with IsoRank , MI-GRAAL, Graemlin and BLAST approach. The dataset is obtained from IntAct database(Aranda *et. al.* 2010). *PINALOG* provides more protein pairs with higher function similarity than IsoRank. This is because of the combination of sequence, function and network neighborhoods in the seed-and extension approach of PINALOG. On the other hand, MI-GRAAL, that uses an integrative approach using sequence, function and topology information, obtains an alignment with poor function similarity between mapped pairs, less than IsoRank and a lot less than PINALOG.

CHAPTER 4

MULTIPLE ALIGNMENT OF PROTEIN INTERACTION NETWORKS VIA THREE-INDEX ASSIGNMENT METHOD

4.1 Preface

A large amount of data on protein interactions is available which has motivated the researchers to compare the networks of different species. The alignment of bio-molecular networks is used for understanding interactions in the networks of different species. Comparing networks allows us to identify conserved functional modules, predict protein functions, validate protein interactions, predict protein interactions or discover protein complexes. To get good results to all of the above advantages we need to formulate an alignment method that is accurate and efficient. Many researchers have successfully developed alignment methods for aligning two networks; extending the alignment to more than two networks becomes difficult as the PPI networks are too large and the complexity increases at a very high rate. Various methods have been developed for multiple alignment of PPI networks. In this chapter we propose a method for aligning multiple PPI network using the solution of Three-Index assignment problem given by (Huang *et. al.*, 2006).As mentioned in the previous chapter the proposed method is an extension of a pairwise network alignment method PINALOG.

.

4.2 Proposed Method

In this section we describe a method for aligning three PPI networks using the methodology followed by PINALOG. PINALOG uses Hungarian algorithm to find maximum match between the proteins of two species. Hungarian algorithm is a solution

to classic two dimensional assignment problem (AP2). Extension of two-dimensional assignment problem is called a multi-dimensional assignment problem. A multi-dimensional assignment problem also referred to as multi-index assignment problem is considered to be a *NP-Hard* problem. Few algorithms have been proposed for multi-index assignment problem but most of them focus on the three-index form of the problem. This is because of the huge computational complexity of the multi-dimensional form that is N-dimensional assignment problem (AP-N). In this thesis we use a solution to Three-Index Assignment problem (Huang *et. al.*, 2006) in order to find an optimal match between the proteins of three species.

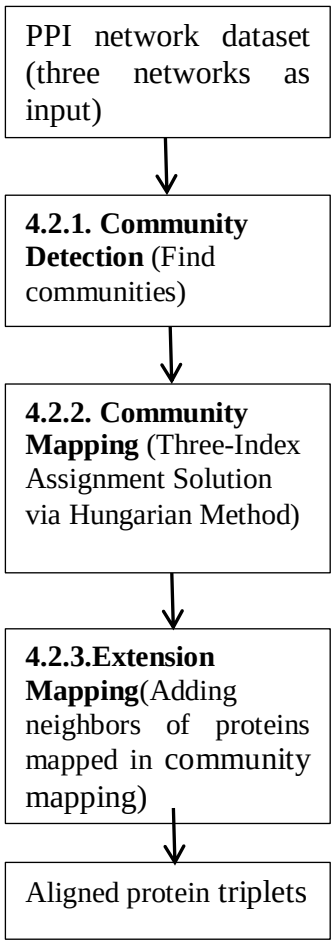


Figure 4.1: Summary of proposed method

Using PINALOG methodology as a reference, the above figure shows the steps involved in alignment of three PPI networks.

4.2.1 Community Detection

As discussed in previous chapter this step identifies the highly connected sub-networks of the input PPI networks. The mapped proteins resulting from aligning protein interaction networks helps us in predicting protein complexes present in the networks and also to identify functions of the proteins present in these networks. It is known that the highly connected sub-networks in PPI networks are formed by protein complexes or functional modules. These identified dense sub-networks are said to be enriched with biological function. Hence, it is better to find the highly connected sub-graphs of these networks and align them first. Many algorithms have been proposed to detect these sense sub-networks in PPI networks called communities. A community is defined as a group of proteins that are more closely associated with themselves than with the rest of the network (Figure 3.10). The process of finding communities is referred to as clustering. Many clustering methods have been proposed .In our method we use quite popular clustering method called CFinder (Palla *et.al.*, 2005) which helps in locating overlapping groups of dense sub-networks in the PPI networks. This method finds overlapping communities in the networks using Clique Percolation Method.

4.2.1.1 Clique Percolation Method

Communities are usually defined as dense parts of networks. Majority of the community detection approaches separate these regions from each other by a relatively small number of links in a disjoint manner. However, in reality communities may even overlap as well. If overlapping takes place, a node in the overlap are considered as members of more than one community. CPM allows in identifying the community overlaps based on link-density.

In this approach a community is built up from adjacent blocks of the same size k. These blocks also called as cliques is a maximum complete sub-graph in which all the nodes are connected to each other thus having the highest possible density. The cliques consist of k members where each of the k members of the k-clique is linked to every other member. Two blocks are considered adjacent if they overlap with each other as strongly as possible, i.e., if they share $k - 1$ nodes. Note that removing one link from a k-clique leads to two adjacent $k - 1$ cliques sharing $k - 2$ nodes. The k parameter can be chosen according to the need of the user (suggested value is between 4 and 6). The figure 4.2 (Tang *et. al*., 2010) shows an example of CPM. In this method a block can be a part of only one community; however, the nodes may belong to several communities at the same time.

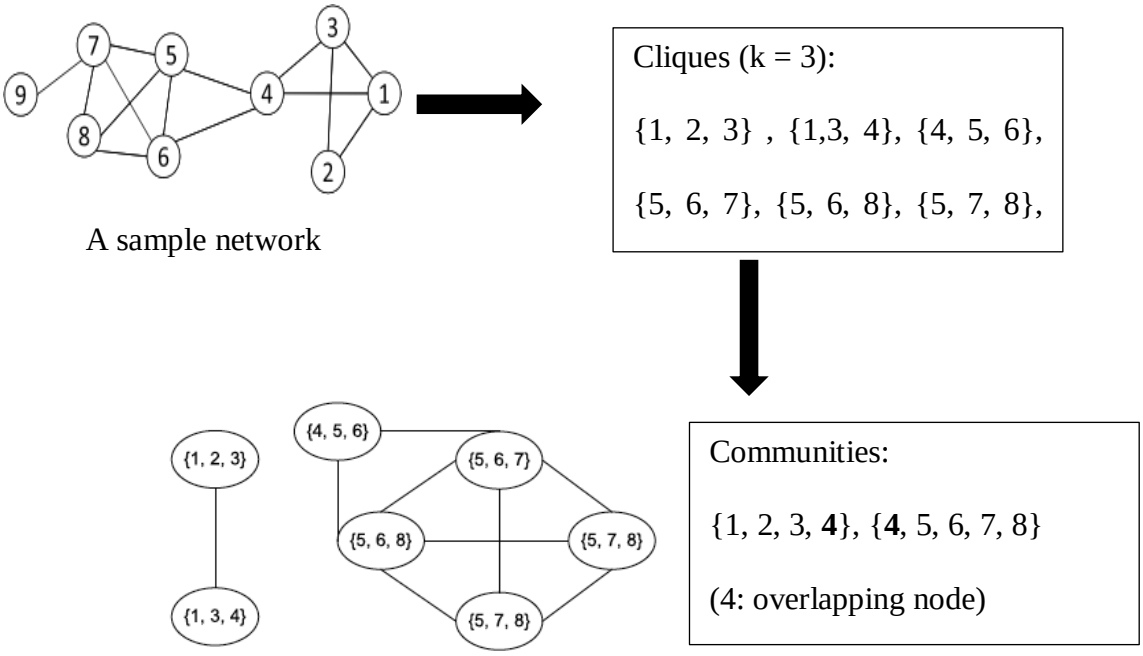


Figure 4.2: Example of clique percolation method

4.2.2 Community Mapping

After the communities have been detected using CFinder this step maps the communities having the highest similarity score and later extracts the mapped proteins in those mapped communities. As discussed earlier, in PINALOG the mapping of communities/proteins is done using Hungarian method. It is an optimization technique that assigns the proteins of two species in an alignment by maximizing the total score in polynomial time. PINALOG is a pairwise network alignment method and hence it uses Hungarian algorithm which deals with two-dimensional assignment problem. Since our algorithm deals with multiple alignment of protein interaction networks we need to extend from two-dimensional assignment method to a multi-dimensional (index) assignment method. A very few algorithms have been proposed for multi-dimensional

assignment problem. Multi-dimensional problem is considered to be NP-Hard and cannot be solved in polynomial time. The main focus of our thesis is to align networks of three species (multiple alignment) thus in order to get the optimization result we need a solution for three-index assignment problem (AP3). In order to match proteins in the three networks we need to compute the similarities between these proteins and obtain similarity matrix. In the following sub-section we discuss in detail the calculation of similarity scores between the proteins of different species used in community mapping.

4.2.2.1 Scoring Scheme

For any alignment the calculation of node scores constitutes an important part as proteins in the networks are matched on the basis of their corresponding score (i.e. similarity). Many algorithms use sequence similarity between the proteins in order to match them. There is no valuation showing that the quality of alignment is dependent on the scoring scheme but we see in PINALOG that including functional similarity in the scoring scheme helps in yielding more matched proteins with higher functional similarity and fewer matched proteins with low functional similarity. The sequence similarity helps us in revealing orthologous relationships between the species but they do not indicate functional similarity. In most alignment methods either sequence similarity or the network topology of the input networks is used. Very few algorithms include functional similarity of proteins. Not having functional similarity as a part of the alignment process may result in matched proteins that have no similarity. The recently developed method MI-GRAAL (Kuchaiev *et. al.*, 2011) presents a global alignment algorithm in which they

use information such as topological features, sequence similarity and functional similarity.

In our alignment method the similarity score is defined as a weighted sum of sequence similarity and functional similarity between the proteins.

$$s(a_i, b_j) = \theta\, s_{seq}(a_i, b_j) + (1 - \theta) s_{func}(a_i, b_j) \quad \textbf{(4.1)}$$

Here, θ depicts the closeness between the proteins of two species.It is relative weighting between the functional similarity and sequence similarity. The value of θ is calculated using the number of reciprocal Blast hits between the protein sequences of the species.

$$\theta = 1 - \frac{R}{M + N} \quad \textbf{(4.2)}$$

Here, M and N are the size of two input networks and R is the number of reciprocal Blast hits which has a high value if two species are very close. The main purpose if including θ to our alignment is to provide a balance between functional and sequence similarity.The value of $\theta = 0.5$ shows that two species are very close.

A. Sequence Similarity

Like aligning networks we can find similar regions within the networks in the same way sequence similarity between proteins is found using the method called sequence alignment. Sequence alignment arranges the sequences of proteins to identify the regions of similarity resulting from functional, structural or evolutionary relationships between the protein sequences. We use the Basic Local Alignment Search Tool (BLAST) to find

similarity between proteins of different species. The sequence similarity of two proteins a_i and b_j is given by the equation.

$$\boldsymbol{s_{seq}}(a_i, b_j) = \frac{S(a_i, b_j)}{\sqrt{S(a_i, a_i)\, S(b_j, b_j)}} \quad \textbf{(4.3)}$$

$S(a_i, b_j)$ is the BLAST bit score value when aligning a_i and b_j.The protein pairs with $E - value < 10^{-5}$ are used to calculate sequence similarity.

B. Functional Similarity

The functional similarity $\boldsymbol{s_{func}}(a_i, b_j)$ of two proteins is calculated by the method proposed by Schlicker *et al.,* 2006 which uses functional similarity between gene products to compare gene annotations. Gene Ontology (GO) provides a standard vocabulary of functional terms, and allows annotation of gene products with one or more descriptive terms. GO is divided into three parts: *molecular function, biological process and cellular component.* For calculating functional similarity we us biological process (BP) and molecular function (MF) as cellular component (CC) annotation of different species is not similar and thus CC terms cannot be used. The ontologies are independent of each other and thus a gene product can be annotated with terms from all the ontologies.. A directed acyclic graph (DAG) (Figure 4.3) is used to depict all three ontologies where nodes of the graph represent the terms/concept which consists of a certain amount of information and the edges (links) represent the relationship between the terms. Nodes that are close to each other represent similar concepts. There are two kinds of semantic relationships between the nodes; "$is - a$" and "$part - of$" links. "$is - a$" is a simple class-subclass relation, where A "$is - a$" B means that A is a subclass of B.

"$part - of$" is a partial ownership relation where C part-of D means that whenever C is present, it is always a part of D, but vice-versa is not true.

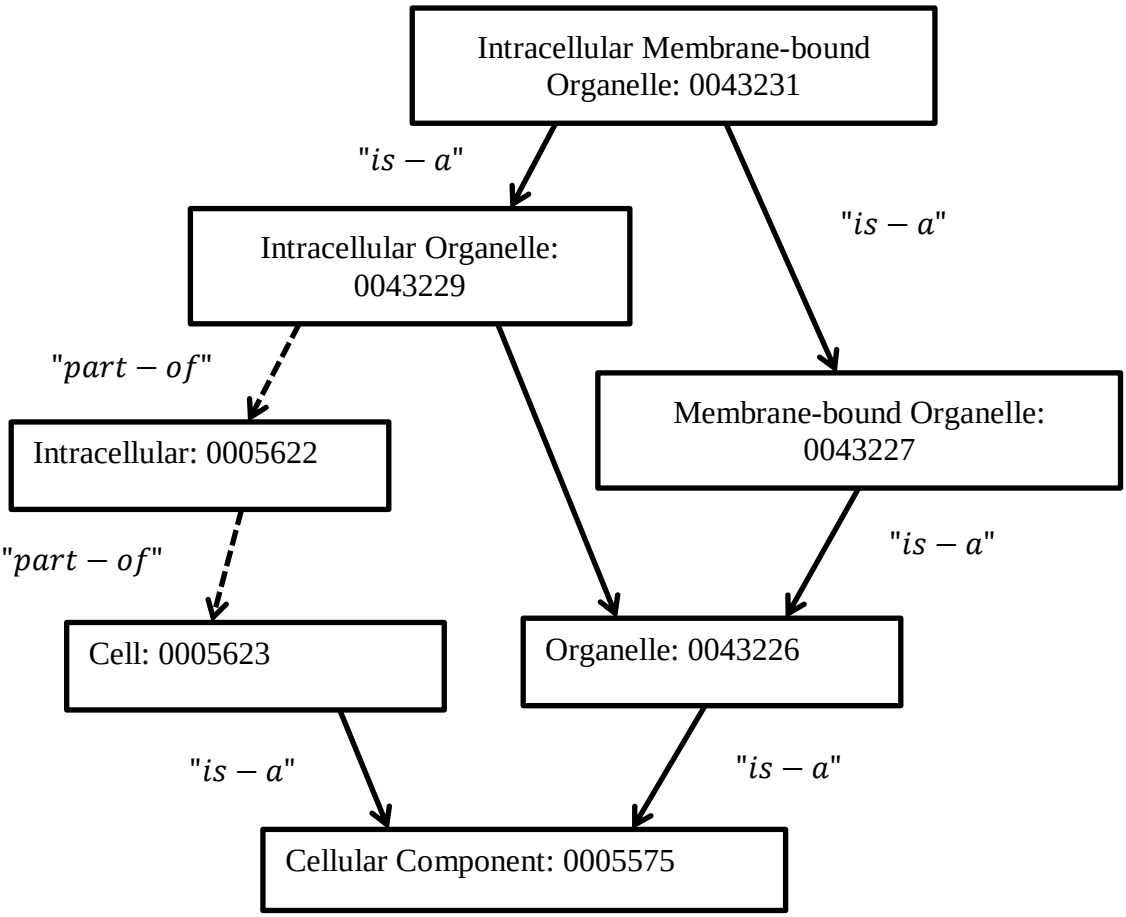


Figure 4.3: DAG for Intracellular membrane-bound organelle: 0043231(Wang *et. al.*, 2007).

Calculating similarity between two concepts is based on the amount of information they have in common. The protein functional similarity proposed by (Schlicker *et al,* 2006) is calculated based on semantic similarity by modifying the method described by (Lin *et. al,*.1998*) and (*Resnik *et. al*., 1995b*)* referred to as node based (information-content) similarity measure. Both similarity measures rely on the concept of information content.

The information content of a GO term is calculated using a probability function p which estimates the probability of occurrence of the concept in a large text corpus. Because of the hierarchical structure, the concept present higher in hierarchy absorbs the ones which are lower in the hierarchy making probability a monotonic function. Thus the value of probability increases as we move up in hierarchy.

Semantic Similarity Methods

Lin *et. al,*.1998 defines the similarity as the ratio of amount of information needed to state the commonality of the two concepts and the information needed to fully describe the two concepts whose similarity we need to find. However, Resnik *et. al*., 1995b uses information content (IC) to define the conceptual similarity between the two concepts. The similarity between concepts is based on the amount of information the share. Thus IC of a concept c is calculated as $IC(c) = -log_{10}\, p(c)$ where $p(c)$ is the probability of encountering an instance of a concept c. The following equations show the semantic similarity measure described by Lin and Resnik respectively.

$$sim_{LIN}(c_1, c_2) = \max_{c \in S(c_1,c_2)} \left(\frac{2 \log p(c)}{\log p(c_1) + \log p\, c_2)}\right) \quad \textbf{(4.4)}$$

Where $S(c_1, c_2)$ is a set of common ancestors of concepts (c_1, c_2).The value of this similarity ranges from 0 to 1.

$$sim_{RESNIK}(c_1, c_2) = \max_{c \in S(c_1,c_2)}(-\log p(c)) \quad \textbf{(4.5)}$$

Where $S(c_1, c_2)$ is a set of common ancestors of concepts (c_1, c_2). The minimum similarity value is zero but there is no upper bound for this method.

Thus given two GO terms (c_1, c_2), the relevance semantic similarity score used by Schlicker *et al.,* 2006 in his method is a modification of Lin's similarity score

$$sim_{Relevance}(c_1, c_2) = \max_{c \in S(c_1,c_2)} \left(\frac{2 \log p(c)}{\log p(c_1) + \log p\, c_2)} \ (1 - p(c)) \right) \quad \textbf{(4.6)}$$

The value of this score also ranges between 0 and 1.

Schlicker's functional similarity method

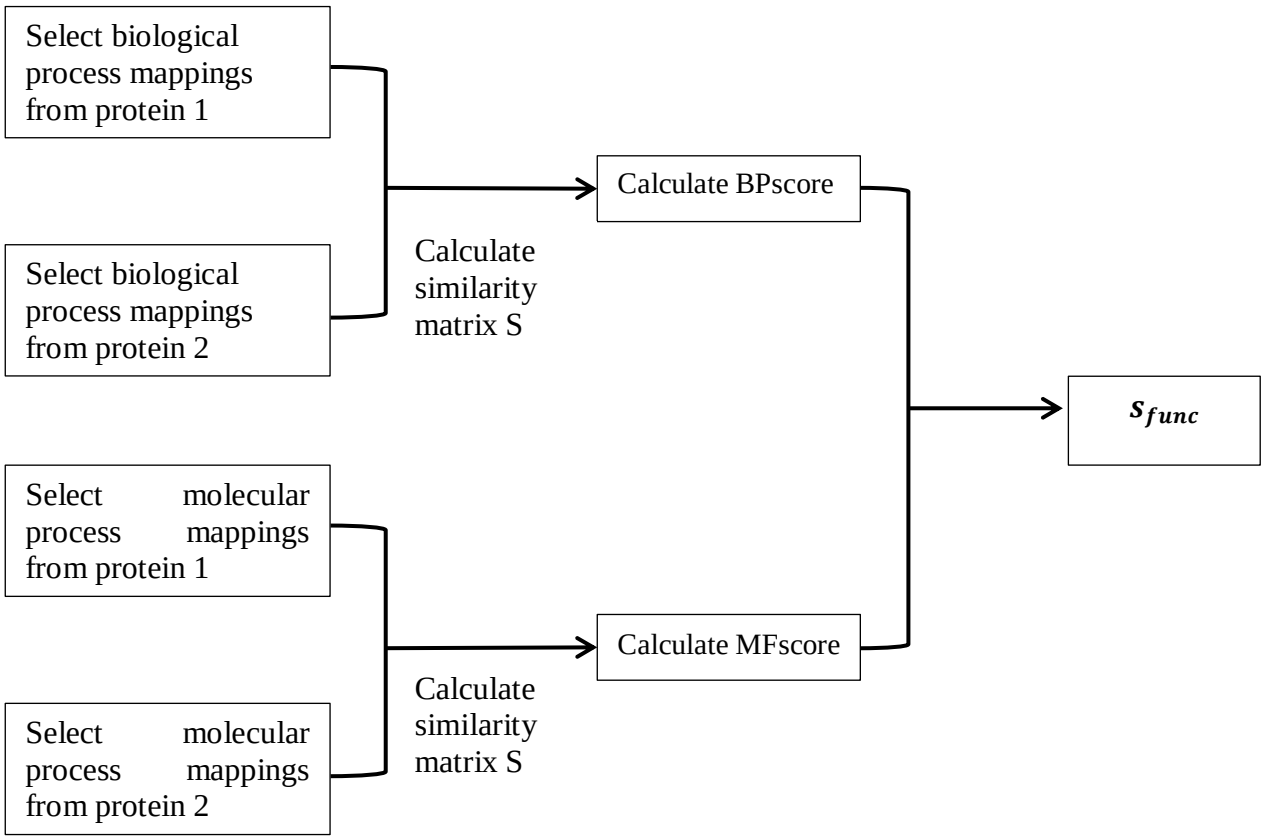


Figure 4.4: Diagram showing calculation of functional similarity for two proteins.

- The first step is to compute a similarity matrix S for the two proteins say a_i and b_j by pairwise comparison of their GO mappings. The mappings to different ontologies (*molecular function, biological process)* are calculated separately.

Consider two proteins a_i and b_j associated with the sets G^a and G^b of GO terms. The similarity pairwise values of mapping G_i^a of protein a and mappings G_i^b of mappings of protein b are calculated using $sim_{Relevance}$.

$$S = \begin{bmatrix} sim(G_1^a, G_1^b) & sim(G_1^a, G_2^b) & \cdots & sim(G_1^a, G_M^b) \\ sim(G_2^a, G_1^b) & sim(G_2^a, G_2^b) & \dots & sim(G_2^a, G_M^b) \\ \vdots & \dots & \ddots & \vdots \\ sim(G_N^a, G_1^b) & sim(G_N^a, G_2^b) & \cdots & sim(G_N^a, G_M^b) \end{bmatrix}$$

The algorithm assigns the best hit (value) to every row and column. The best hit is defined as the highest similarity score. This score represents the functional similarity between the proteins. To find the best hits we find maximum values in rows (row maxima) and columns (column maxima).

$$rowScore = \frac{1}{N}\sum_{i=1}^{N} \max_{1 \le j \le M} sim(G_i^a, G_j^b) \quad \textbf{(4.7)}$$

$$columnScore = \frac{1}{M}\sum_{i=1}^{M} \max_{1 \le j \le N} sim(G_i^a, G_j^b)$$

The value of $rowScore$ and $columnScore$ ranges between 0 and 1.The maximum $Gscore_{max}$ is calculated using the equation

$$Gscore_{max} = \max\{columnScore, rowScore\} \quad \textbf{(4.8)}$$

- This score is calculated both for molecular function ($MFscore$) and biological process ($BPscore$). Thus the final functional similarity score is calculated by combining $MFscore$ and $BPscore$:

$$S_{func} = \frac{1}{2}\left[\left(\frac{BPscore}{\max BPscore}\right)^2 + \left(\frac{MFscore}{\max MFscore}\right)^2\right] \quad \textbf{(4.9)}$$

4.2.2.2 Three-Index Assignment Problem

The AP3 is an optimization problem on a complete tripartite graph. The cost of choosing triangle $(i, j, k)\ is\ c_{ijk}$. The objective of AP3 applied in our method is to choose *"N"* disjoint triangles *(i, j, k)* so that the total cost is maximized. The 0-1 programming model for AP3 (Huang *et. al.*, 2006) is:

$$max\ \sum_{i=1}^{N}\sum_{j=1}^{N}\sum_{k=1}^{N} c_{ijk}x_{ijk} \quad \textbf{(4.10)}$$

subject to

$$\sum_{j=1}^{N}\sum_{k=1}^{N} x_{ijk} = 1 \qquad \forall\, i \in N$$

$$\sum_{i=1}^{N}\sum_{k=1}^{N} x_{ijk} = 1 \qquad \forall\, j \in N$$

$$\sum_{i=1}^{N}\sum_{j=1}^{N} x_{ijk} = 1 \qquad \forall\, k \in N$$

$$x_{ijk} \in \{0,1\} \quad \forall \{i,j,k\} \in I\ x\ J\ x\ K$$

Where $\{x_{ijk}\}_{N\ x\ N\ x\ N}$ - Resulting binary matrix, where $\boldsymbol{x_{ij}}$ = 1 if all the elements are assigned and $|i| = |j| = |k| = N$ are disjoint sets.

Clearly AP3 is an extension of Hungarian algorithm which is a solution to AP2 problem. Hungarian algorithm provides a perfect matching of complete bipartite graph. Mathematically matching in AP2 is a bijective mapping of a finite set say $N = \{1, 2, 3, \ldots . n\}$ into itself i.e. permutation $\emptyset$ is matched to some $j = \emptyset(i)$ to each $i \in N$. A permutation $\emptyset$ of set $N = \{1 \ldots \ldots n\}$ represents a permutation matrix $P_{\emptyset} = x_{ij}$ where $x_{ij} = 1$ for all $j = \emptyset(i)$ and $x_{ij} = 0$ where $j \neq \emptyset(i)$. Thus the matrix $P_{\emptyset}$ represents the adjacency matrix which fulfills the conditions given in Equation 3.1(see Figure 4.4 (Bukard *et. al.*, 1999)).

$$\emptyset = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 3 & 2 & 4 & 1 \end{pmatrix}$$

$$P_{\emptyset} = \begin{pmatrix} 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \end{pmatrix}$$

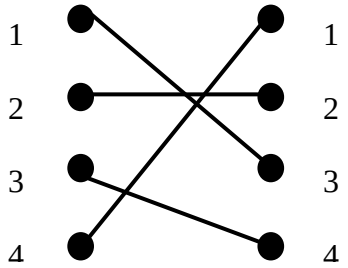


Figure 4.5: Diagram representing mathematical representation of an assignment in AP2.

Using the above explained notation we can represent a solution to AP3 problem using three permutations. The permutation representation of AP3 is formulated as

$$\max \sum_{i=1}^{n} c_{I(i),\ p(i),\ q(i)} \qquad \textbf{(4.11)}$$

With $I, p, q \in \pi_N$ where π_N is a set of all permutations on the set of integers $N = \{1, 2, 3 \ldots \ldots \ldots n\}$. If we fix one permutation say the index permutation ($I(i) = i$) the solution to AP3 problem can be represented by using pair of permutations (p, q).

$$\mathrm{C(p, q)} = \max \sum_{i=1}^{n} c_{i,\ p(i),\ q(i)} \qquad \textbf{(4.12)}$$

- **Method**

We know that AP3 is a NP-hard problem. Thus we solve this problem by projecting the three-dimensional onto the two-dimensional problem. As stated earlier a solution to AP3 consists of two permutations $p\ and\ q$. However, a solution to AP2 consists of only one permutation say q. We assume an initial solution to our AP3 problem as (p, q) and fix the index permutation. We create the bipartite graph based on the following equation

$$d_{ij} = c_{i,\ p(i),\ j} \quad \forall\, j \in 1, 2, \ldots \ldots n \qquad \textbf{(4.13)}$$

Now our objective is to satisfy the equation

$$\max_{p,q \in \pi_N} \sum_{i=1}^{n} c_{i,\ p(i),\ q(i)} = \max_{q \in \pi_N} \sum_{i=1}^{n} d_{i,q(i)} \qquad \textbf{(4.14)}$$

Here, we fix permutation p, the optimization of q becomes an AP2 problem.

Objective: Our idea is to optimize one permutation subject to the other permutation being fixed. Below we discuss an example (Huang *et. al.*, 2006) showing how our algorithm works.

- **Explanation**

  - Consider graphs of three species having four nodes each. The figure 4.5 shows a tripartite graph in which each shape represents different species. The permutation p is matching between graph 1 and 2 and q between 2 and 3. We fix the index permutation and consider a random initial assignment of the tripartite graph. As mentioned above we aim to optimize one permutation at a time keeping the other permutation fixed. By doing this we attain an overall optimized result. The cost matrix used in order to obtain maximum matching between the nodes of three graphs consist of the similarity

score obtained from Equation 4.1 between proteins of each specie. Since we want to match proteins which have maximum similarity our algorithm performs maximum weight optimization on the three input graphs.

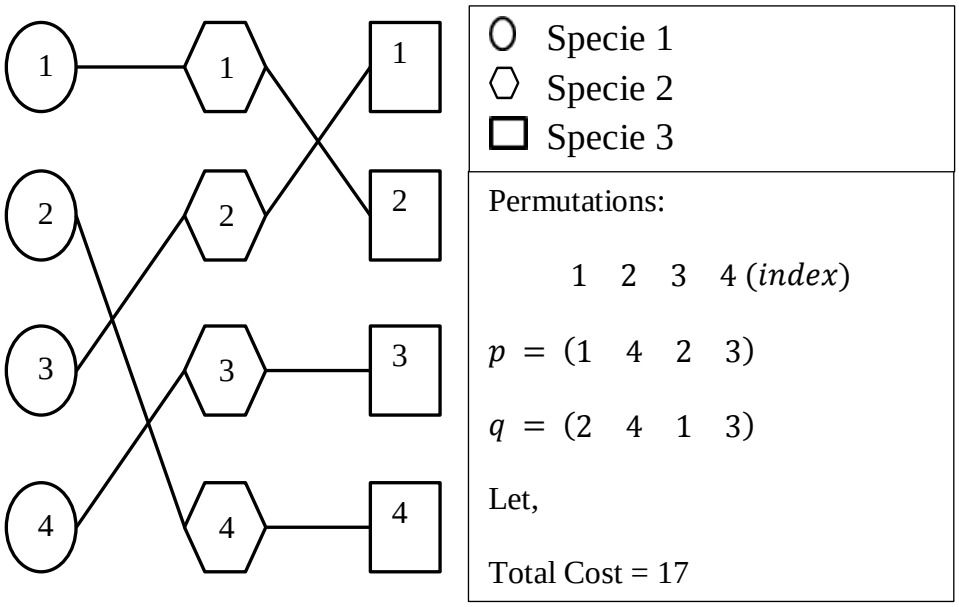


Figure 4.6: Random initial assignment of three graphs.

- **Optimize permutation q:** As mentioned earlier we project our 3-dimensional problem onto 2-dimensional. Hence, we now construct a bipartite graph combining the nodes of specie 1 and 2 as shown in Figure 4.6 using the Equation 4.13 and optimize the corresponding bipartite graph using Hungarian algorithm. This algorithm provides us with the maximum matching of the newly constructed bipartite graph. After using the Hungarian algorithm we check if the value of total similarity score (cost) has increased. If the total cost value increases we change the values of permutation q in accordance to the new assignment as well as update the cost value.

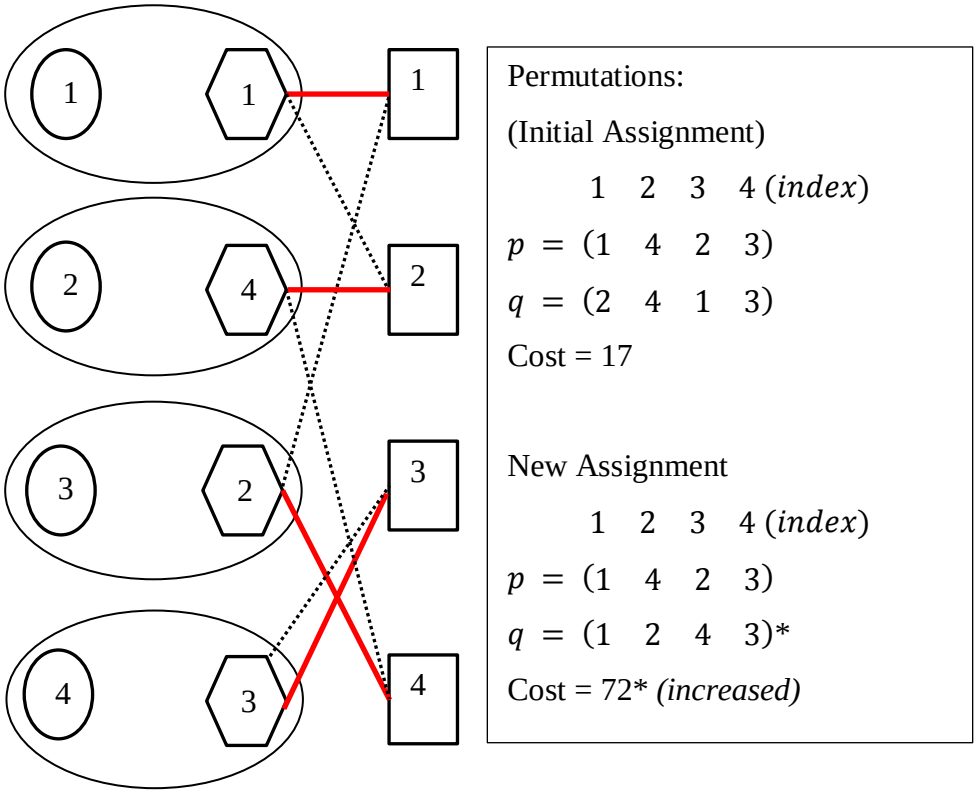
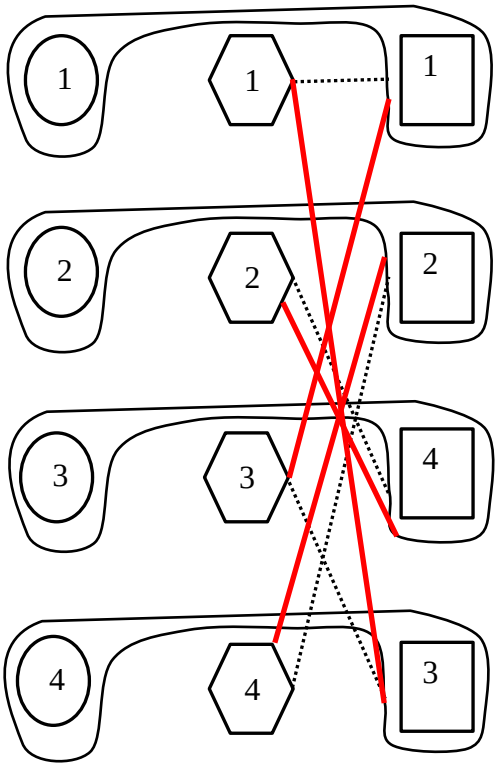


Figure 4.7: Diagram showing optimization of permutation q

- ***Optimize permutation p-*** We know the three index assignment problem consists of two permutations. The previous step optimizes the permutation q using Hungarian Algorithm. Similarly we now optimize permutation p which is an assignment between nodes of species 1 and 3. We again construct a corresponding bipartite graph as shown in Figure 4.7 and optimize it by applying the Hungarian Algorithm this graph. We check if there is any increase in the cost value. Since we want to maximize the total cost, we update the permutation values and the value of total cost if the cost value is greater than the previous one.



Permutations:

(Updated Assignment)

$1 \quad 2 \quad 3 \quad 4\ (index)$

$p = (1 \quad 4 \quad 2 \quad 3)$

$q = (1 \quad 2 \quad 4 \quad 3)$

Cost = 72

New Assignment

$1 \quad 2 \quad 3 \quad 4\ (index)$

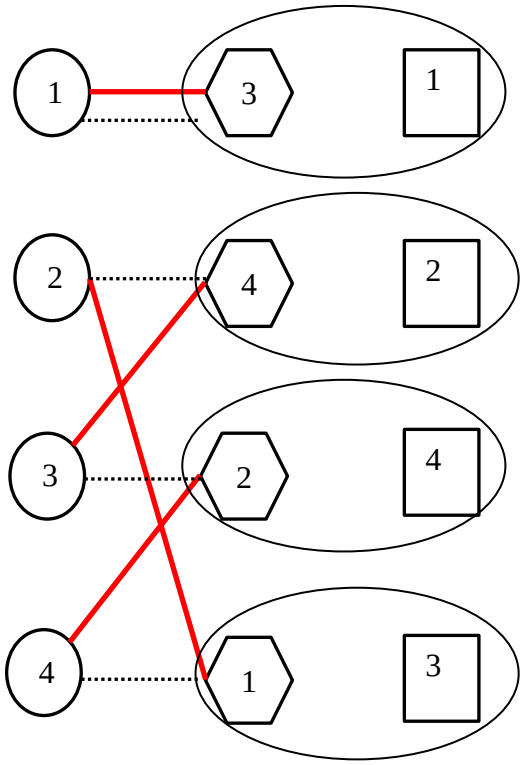
$p = (3 \quad 4 \quad 2 \quad 1)$*

$q = (1 \quad 2 \quad 4 \quad 3)$

Cost = 120* *(increased)*

Figure 4.8: Diagram showing optimization of permutation q

- ***Optimize index permutation***-



Permutations:

(Updated Assignment)

$1 \quad 2 \quad 3 \quad 4\ (index)$

$p = (3 \quad 4 \quad 2 \quad 1)$

$q = (1 \quad 2 \quad 4 \quad 3)$

Cost = 120

New Assignment

$1 \quad 3 \quad 4 \quad 2\ (index)$

$p = (3 \quad 4 \quad 2 \quad 1)$

$q = (1 \quad 2 \quad 4 \quad 3)$

Cost = 177* *(increased)*

Figure 4.9: Diagram showing optimization of index permutation *i*

After optimizing both the permutations we finally optimize the index permutation to get the final optimal assignment. We construct the bipartite graph by combining nodes od species 2 and 3and optimize it by applying the Hungarian Algorithm on the bipartite graph in Figure 4.8. After checking the cost value and updating the permutation values if needed we get the final matched triplets of proteins having maximum similarity among them.

In the above example we illustrate the steps used for aligning three PPI networks. As mentioned in section 3.3.1.2 of PINALOG the community mapping step consist of two parts.In order to find most similar communities of the three input networks we first have to match the proteins present in those communities to obtain community similarity matrix. After getting the mapped communities we extract the triplets of proteins one from each specie, which have the maximum similarity between them. These proteins are referred to as core proteins. These core proteins are considered as candidates for the next step of our algorithm. Before moving onto the next step of our algorithm we do a filtering step in which only top 15% of the seed proteins are retained for the extension step.

4.2.2.3 Algorithm

Algorithm: Community Mapping
Input: Set of Communities $\{C_1^A, C_2^A \dots\dots\dots C_i^A\}$ of network A , $\{C_1^B, C_2^B \dots\dots\dots C_j^B\}$ of network B and $\{C_1^C, C_2^C \dots\dots\dots C_k^C\}$ for network C
Other variables:
(A_iB_j): Similarity matrix containing similarity score between proteins present in C_i^A and C_j^B
(B_jC_k): Similarity matrix containing similarity score between proteins present in C_j^B *and* C_k^C
(A_iC_k): Similarity matrix containing similarity score between proteins present in C_i^A and C_k^C
$(C_i^AC_j^B)$ = Community similarity matrix of networks A and B
$(C_i^BC_j^C)$ = Community similarity matrix of networks B and C
$(C_i^AC_j^C)$ = Community similarity matrix of networks A and C
S_{ij}^{AB} = The total optimized score matched in community I of network A and community j of network B respectively after applying Hungarian algorithm.
S_{jk}^{BC} = The total optimized score matched in community j of network B and community k of network C respectively after applying Hungarian algorithm.
S_{ik}^{AC} = The total optimized score matched in community i of network A and community k of network C respectively after applying Hungarian algorithm.
Output: Set of protein triplets matched

Begin

1. For each community in C_i^A and in C_j^B DO
   S_{ij}^{AB} = Hungarian (A_iB_j) // Obtain maximum score between all the communities of network A and B
   $(C_i^AC_j^B)$: = S_{ij}^{AB} // Add the total matched score to construct community similarity matrix of network A and B

   For each community in C_j^B and C_k^C DO
   S_{jk}^{BC} = Hungarian (B_jC_k) // Obtain maximum score between all the communities of network A and B
   $(C_i^BC_j^C)$: = S_{jk}^{BC} // Add the total matched score to construct community similarity matrix of network B and C

   For each community in C_i^A and in C_j^B DO
   S_{ik}^{AC} = Hungarian (A_iC_k) // Obtain maximum score between all the communities of network A and B
   $(C_i^AC_j^B)$: = S_{ik}^{AC} // Add the total matched score to construct community similarity matrix of network A and C

2. Three_Index_Matching ($C_i^AC_j^B$, $C_i^BC_j^C$, $C_i^AC_j^C$) // refer Figure 4.10

End

Figure 4.10: Algorithm for community mapping

Algorithm: Three_Index_Matching
Input: Community matrix of three species ($C_i^A C_j^B$, $C_i^B C_j^C$, $C_i^A C_j^C$)
Other variables:
flag: A variable to check the quit condition.
Score: The total optimized score of the assignment.
(p, q): two permutations of the AP3 assignment.
Output: Set of matched proteins one from each specie.
Begin

1. Consider (p, q) = initial solution.
2. Set flag = false.
3. While flag = false DO
   flag = true.
   For 1-2 part of solution DO
   - Construct bipartite graphs using equation 4.12
   - Optimize by applying Hungarian Algorithm
   - If Score increases
     Flag = false.
     End if

   End for
   End while

End

Figure 4.11: Algorithm for Three-Index Assignment problem

4.2.3 Extension Mapping

In the community mapping process we include the sequence as well as functional similarity to our algorithm. Extension mapping aims at including the neighborhood similarity of the networks being aligned. The seed proteins obtained in the previous step are used to extend the alignment over the whole network. The first and second neighbors of the proteins in the core are used in this step. The set of all proteins separated by one interaction and proteins separated by two interactions of a_i in A and b_j in B are denoted by $N(a_i)$ and $N(b_j)$. Let $d(a_k, a_i)$ denote the distance between a_k and a_i in a network. The similarity between a_i and b_j in extension mapping is then defined as

$$s_{ext}\left(a_i, b_j\right) =$$

$$s\left(a_i, b_j\right) + \sum_{\substack{a_k \in N(a_i) \\ b_l \in N(b_j) \\ (a_k,\ b_l) \in core}} \frac{1}{(d(a_k, a_i) + 1)(d(b_l, b_j) + 1))}\ s(a_k,\ b_l) \quad \textbf{(4.15)}$$

These proteins are again aligned using Three-Index assignment solution described in the previous section. The example of extension mapping is given in Figure. This step is repeated until no more proteins can be added into the core.

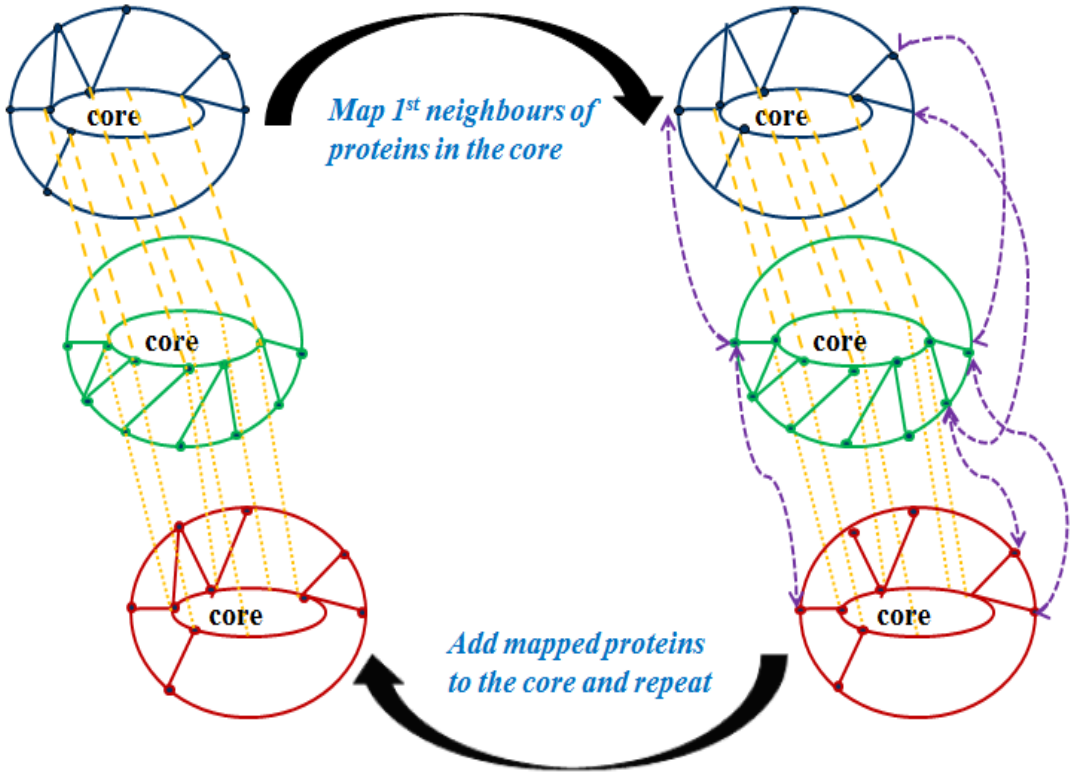


Figure 4.12: Extension mapping from core proteins.

The final alignment may result in many-to-many mappings as CFinder finds overlapping communities, in which one protein may be mapped to more than one protein in the other networks. Thus, we find alignment between three PPI networks using these three steps.

CHAPTER 5

EXPERIMENTS AND RESULTS

5.1 Preface

This section discusses the experiments performed to evaluate our proposed method in terms of effectiveness and accuracy of the alignment of three protein interaction networks and execution time analysis.

.

5.2 Dataset used for Evaluation

To test the correctness of our algorithm we use publically available dataset provided by PINALOG. It consists of nine protein-protein interaction networks. And can be downloaded from the website *http://www.sbg.bio.ic.ac.uk/~pinalog/downloads.html*. The dataset is a collection of proteins and the interactions present in the PPI networks shown in the table. It also consists of Blast score file which consists of sequence similarities between the proteins of all networks and also the GO annotation file. Since our algorithm is based on PINALOG we download this dataset.

Table 5.1: PINALOG Dataset

PPI Network	**Number of Proteins**	**Number of Interactions**
Bacterium (*Escherichia coli*)	2817	13841
Fruit fly (*Drosophila melanogaster*)	8366	25611
Flowering plant (*Arabidopsis thaliana*)	2651	5236
House mouse (*Mus musculus*)	2897	4372

Human (*Homo sapiens*)	8994	34935
Street rat (*Rattus norvegicus*)	1150	1307
Round worm (*Caenorhabditis elegans*)	4303	7747
Baker's Yeast (*Saccharomyces cerevisiae*)	5672	49830

The PPI network data file (graphs) consists of two columns of protein names in the species. The BLAST data file contains the result of the all-against- all BLAST results of protein the input species. Each file has a specific format shown in Table 5.2

Table 5.2: File Format of PINALOG Dataset

Field Name	**Description**	**Format**
PPI network Data File	It consists of two columns showing the interaction between the two proteins where each column represents the protein name in the species	$p1 \quad p2$ $p1 \quad p3$ ... ...
BLAST Data File	This file contains the result of the all-against-all BLAST results of proteins in the input species. This includes the BLAST results of the proteins within each species as well as with those in the other species	$p1 \quad p2 \quad Blast_{score}$ $p1 \quad p3 \quad Blast_{score}$ ... ...

We also use dataset used by IsoRank. As we compare our results with the IsoRank we use its dataset available with the IsoRank executable. The dataset consists of eukaryotic PPI networks: H. sapiens (*Human*), M. musculus (*Mouse*), D. melanogaster *(Fly*), C. elegans (*Worm*), and S. cerevisiae (*Yeast*) and Bacterium (*Escherichia coli*)

Table 5.3: IsoRank Dataset

PPI Network	**Number of Proteins**	**Number of Interactions**
Bacterium (*Escherichia coli*)	1821	6848
Fly (*Drosophila melanogaster*)	7518	25829
Mouse (*Mus musculus*)	290	254
Human (*Homo sapiens*)	9635	36381
Worm (*Caenorhabditis elegans*)	2805	4572
Yeast (*Saccharomyces cerevisiae*)	5501	31898

5.3 Evaluation Criteria

We use the PPI networks present in the IsoRank Dataset and the PINALOG dataset to measure the effectiveness and accuracy of our algorithm. In the first step of our algorithm we use a clustering method CFinder to detect communities in the input networks. We know CFinder provides us with overlapping communities. Because of these overlapping communities we get our final alignment with many-to-many mappings. Since, IsoRank generates alignments with one-to-one mapping we reduce our final alignment to a one–to-one mapping. In order to compare our result with IsoRank we reduce our alignment to

a one-to-one mapping by selecting the protein pair with highest similarity $s(a_i, b_j)$ (Equation 4.1) present in many-to-many mapping.

Since there is no gold standard available to compare the results of different alignment methods we use different metrics to see how effective our algorithm is. The effectiveness and correctness of our algorithm is measured on the following criteria.

- NA - The number of matched protein triplets.
- NC - The number of conserved interactions.
- NH – The number of matched protein triplet belonging to the same Homologous groups *(Wheeler et al., 2005).*
- NI - The number of interologs *(Walhout et al., 2000.)*
- NF - The number of matched protein triplets with functional similarity > 0.5.

These measures are used identify if our method is useful for extracting relevant biological information from the resulting alignment. To assess the validity of our algorithm:

- We count the number of proteins aligned in three species represented as NA.
- We also calculate the number of conserved edges of these aligned proteins. If two protein nodes forming an interaction in one species have correspondence to two protein nodes which also form an interaction in the other species then the interaction between those nodes is called conserved interaction (NC). Consider nodes $a_1 \, and \, a_2$ proteins in network A and $b_1 \, and \, b_2$ represent proteins in network B. Now to find conserved edges we check if node a_1 is a neighbour of a_2 in the network. Similarly we check if nodes $b_1 \, and \, b_2$ are neighbours. If this condition holds true then we consider $a_1 - b_1$ and $a_2 - b_2$ as conserved edges.

- The next measure we use for evaluating the quality of our algorithm is NI. It counts the number of interologs present in our final alignment. According to (Walhout *et.al., 2005)* if interacting proteins $a_1 \, and \, a_2$ have interacting orthologs $b_1 \, and \, b_2$, then the pair of interactions $a_1 - b_1$ and $a_2 - b_2$ are called interologs (Figure 5.1). Two proteins are said to be orthologous if their BLAST $E - value \leq 10^{-10}$.

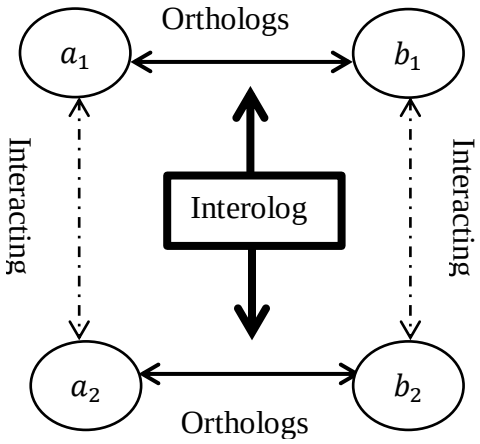


Figure 5.1: Diagram illustrating protein-protein interologs

- The functional similarity between the aligned proteins is also investigated to evaluate whether our alignment methods provides relevant biological information. NF is the measure used which denotes the number of aligned proteins with functional similarity > 0.5.The value of functional similarity used is mentioned in Equation 4.9

- Calculating the number of aligned proteins belonging to the same homologous group which is a common measure for quantifying an alignments quality. NH is the notation used to denote number of Homologene pairs.

5.4 Results

In this section, we describe our results in comparison to the IsoRank (Singh R. *et. al*., 2008) which is a global network alignment method for multiple species. We evaluate both the algorithms using the metrics described above. We compare the results of aligning the three species *Caenorhabditis elegans, Drosophila melanogaster* and *Escherichia coli* present in the IsoRank dataset. The values of NA and NC measure the scale of the alignment. But NA or NC cannot be used to measure the accuracy of an alignment as these metrics may not be biologically relevant. We observe a large difference between NC for our algorithm (2393) and IsoRank (996) which highlights that remarkably different alignments are obtained. We also see that the number of Homologenes (NH) is substantially higher using our algorithm (460 pairs) compared with IsoRank (252 pairs). This is the result of the extension mapping step. This metric shows that our alignment has more effective results than IsoRank.

Also we aligned another set of three species *Caenorhabditis elegans, Drosophila melanogaster, Mus musculus* from the IsoRank dataset see Table 5.4.

Table 5.4: Alignment results of different species from IsoRank dataset. The statistics (Stat) are NA, NC, NF, NH, NI. The ratios NH/NA and NI/NC are also given.

PPI Networks	**Statistics**	**Three-Index Assignment**	**IsoRank**
Worm, fly, bacteria	NA	4191	**4496**
	NC	**2393**	996
	NF	**1990**	840
	NI	**358**	97
	NI/NC	**0.14**	0.09
	NH	**423**	252
	NH/NA	**0.10**	0.05
Worm, mouse, fly	NA	2387	**2922**
	NC	**1127**	579
	NF	**421**	364
	NI	**190**	52
	NI/NC	**0.16**	0.08
	NH	**170**	63
	NH/NA	**0.07**	0.02

We also compare our results with IsoRank using the PINALOG dataset. There are two bottlenecks associated with the IsoRank algorithm. The algorithm requires all vs. all bit scores of BLAST alignments for every protein in the compared organisms. The algorithm requires a number of repetitions for the data to converge. Thus, the algorithm is not fast enough to produce a result in a reasonable time. Because of this reason, we use a small dataset to compare the performances of the two methods.

We also use a set of three species Flowering plant (*Arabidopsis thaliana*), House mouse (*Mus musculus*) and Street rat (*Rattus norvegicus*) from PINALOG dataset to compare the performance of our algorithm. Following table shows the validation metrics of this alignment.

Table 5.5: Alignment results of different species from PINALOG dataset. The statistics (Stat) are NA, NC, NF, NH, NI. The ratios NH/NA and NI/NC are also given.

PPI Networks	**Statistics**	**Three-Index Assignment**	**IsoRank**
Plant, Mouse, Rat	NA	2137	**2322**
	NC	**940**	475
	NF	**1140**	268
	NI	**172**	43
	NI/NC	**0.18**	0.09
	NH	**132**	57
	NH/NA	**0.06**	0.02

Here in the above Table 5.5 we again see that we have significantly high values of NC and NH. The reason for this remarkable difference is because of the combination of sequence, function and topological information in our alignment. IsoRank uses only sequence and topological information to align the networks. Because we add functional information we get more aligned proteins with higher functional similarity and fewer with low functional similarity. Including the functional information we obtain much larger number of functionally similar proteins in the alignment, with 1,140 aligned

proteins having functional similarity score greater than 0.5; which is approximately 40% more than *IsoRank*.

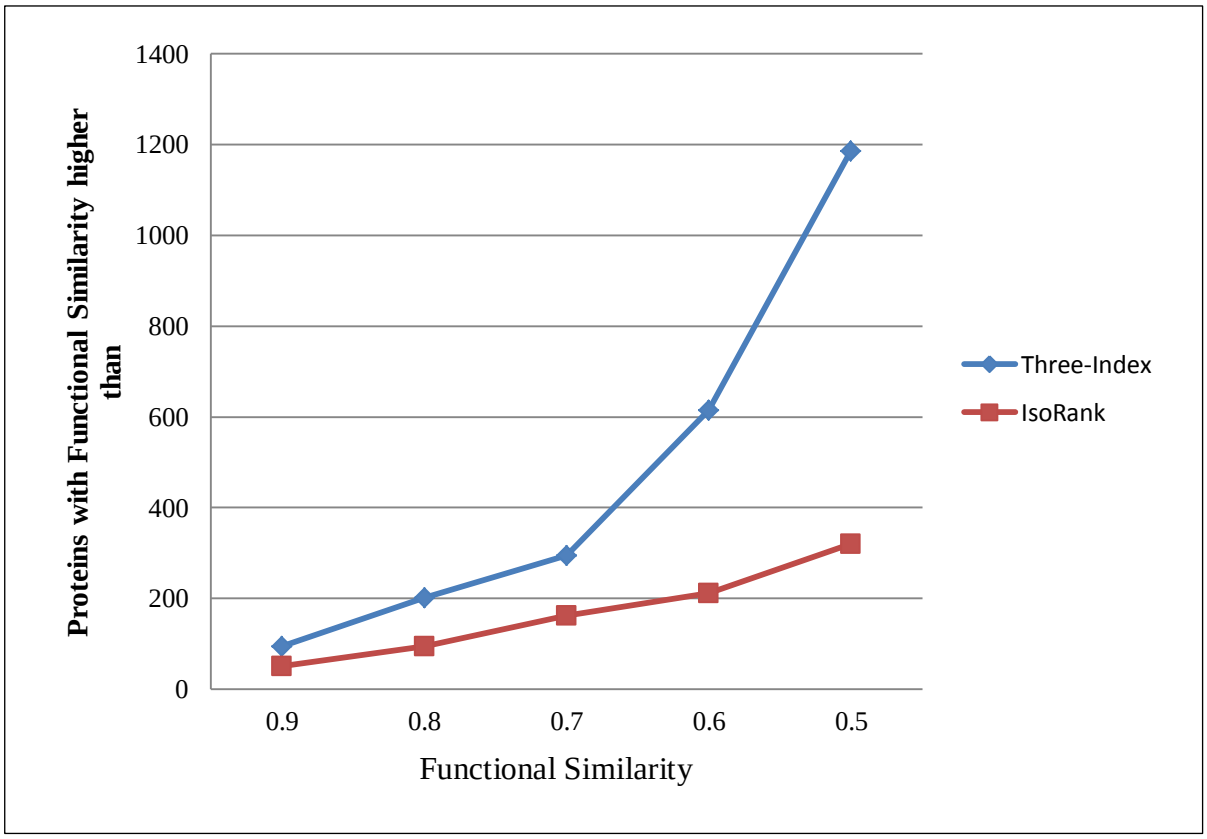


Figure 5.2: Proteins aligned in Worm, Fly and Bacteria with Functional Similarity > 0.5 in comparison with IsoRank

5.5 Runtime Analysis

The computational complexity of the proposed method can be evaluated as sum of the steps performed.

In the first step of the algorithm, the communities are detected using a method CFinder developed by (Palla et. al., 2006). Their CPU time of detecting the communities depends on the structure of the input data very strongly; therefore in general no closed formula

can be given. Thus the complexity of this step is excluded from the execution time of our algorithm

In the second step of the algorithm, we map the communities found in the first step using Hungarian algorithm. We know Hungarian algorithm has a time complexity $O(n^3)$ where n denotes the number of nodes in the bipartite graph. In this step of our proposed method we match the communities of the networks using Hungarian algorithm. Thus the computational complexity of applying Hungarian algorithm to match the communities is $O(C^3)$. Here C represents the maximum number of communities obtained in step 1. In order to match the communities we first need to match the proteins present in them. Thus, the proteins present in two communities are matched with the complexity of $O(C^2 * m^3)$ where m represents the maximum number of proteins in the communities. Hence, community step is executed with the total computational complexity of $O(C^3) * O(C^2 * m^3)$.

The final step of the algorithm is based on extending the alignment by including the neighbors of aligned proteins in the previous networks. The number of iterations for this step is observed to be small limiting to a maximum of 3-4 times. Thus the extension mapping step takes $O(n^3)$ where n is the maximum number of proteins present in the networks.

(Note: The number of proteins matched in each step is less than the total number of proteins present in the input network.)

In the following figure W stands for Worm, F stands for Fly , B stands for bacteria, M – stands for Mouse, A stands for Flowering Plant and R stands for Rat.

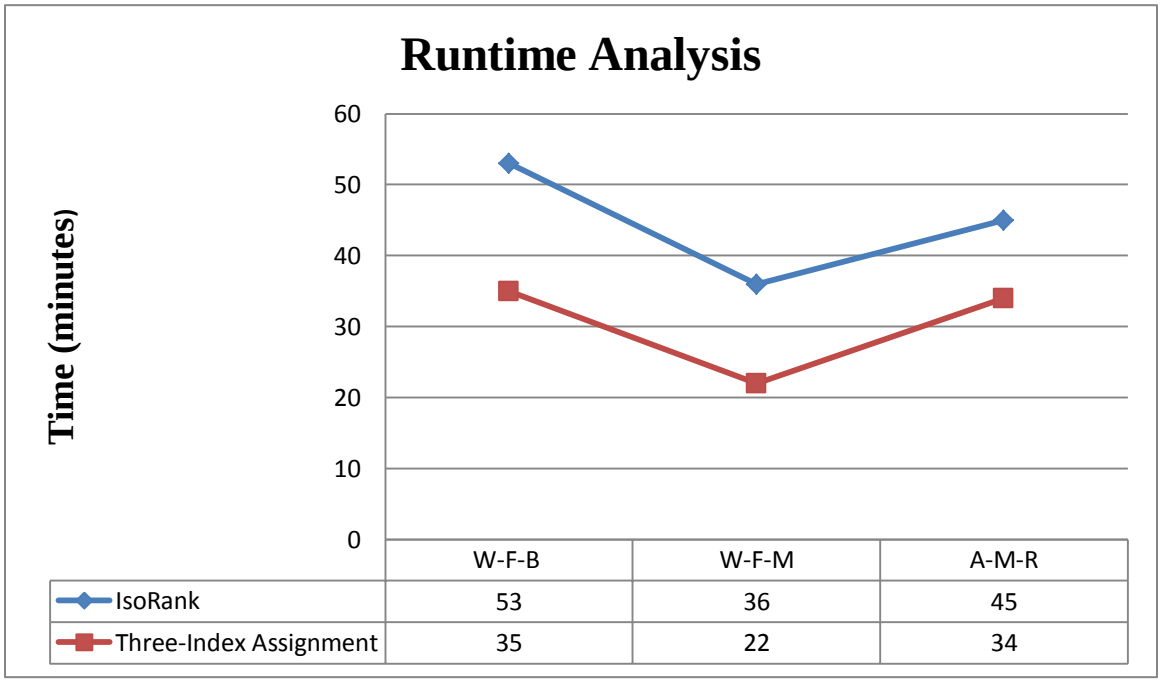


Figure 5.3: Runtime Analysis

Figure 5.3: shows that the run time for our algorithm is less than IsoRank. Hence, our algorithm is more efficient in terms of time complexity as well as revealing relevant biological information (refer Figure 5.2).

CHAPTER 6

CONCLUSION AND FUTURE WORK

In this thesis, an algorithm for aligning three protein interaction networks is proposed. This algorithm uses sequence similarity between the individual proteins of the networks together with the GO annotations of proteins to incorporate functional similarity between the proteins and perform the matching between the proteins of different species. Later, the topological information of the networks is incorporated to get the final alignment. However there are not algorithms proposed for multiple alignment of protein interaction networks.

The proposed algorithm consists of three main steps. The first step is to determine the strongly connected sub-graphs called communities/clusters of proteins in the network. In this step, we use a clustering method called CFinder on the networks being aligned and extract the communities from each network. This way the number of computations required to be performed in the second step of the algorithm is reduced. Also in this step we compute the total similarity function used in the next step by combining both the sequence similarity and function similarity between the proteins of the networks. The second step of the algorithm is to map the communities and obtain the maximum matching between these communities in order to get mapped proteins with maximum similarity. Since we are aligning three networks, we find maximum matching between proteins of these three networks using a method proposed by Huang *et.al* [2004] for three-index (three dimensional) assignment problem using Hungarian algorithm. This

method basically projects the three-dimensional problem on two-dimensional to get maximum matching. In the third step of the algorithm, the alignment is extended to cover all the proteins in the networks. This step uses topological information to extend the matching. The first and second neighbours of proteins obtained in the previous step are considered as candidates for matching. This step is repeated until no more proteins are left for matching.

To evaluate the validity of the aligned proteins, we considered following metrics: (1) NA - the number of aligned protein pairs; (2) NC - the number of conserved interactions; (3)NH - the number of protein pairs belonging to the same Homologene groups (Wheeler et al., 2005); (4) NI - the number of interlogs (Walhout et al., 2000); (5) NF - the number of aligned protein pairs with functional similarity > 0.5 (Schlicker et al., 2006). There is no gold standard available to compare the results of network alignment. Hence we use these measures to check the legitimacy of the alignment methods. The performance of our method is compared with IsoRank which is capable of aligning multiple networks.

The experiments performed on the organisms *Caenorhabditis elegans, Drosophila melanogaster* and *Escherichia coli* from the IsoRank dataset showed that the number of Homologenes (NH) is substantially higher using our algorithm (460 pairs) compared with IsoRank (252 pairs). Also, we see that our alignment produces 3.6 times as many interologs(358) as compared to IsoRank(97). This signifies that our algorithm finds more interologs between species, which along with conserved interactions, might contribute to

the functional similarity of protein interaction networks across species. Since there is a significant difference in the values of NI and NC when we compare both the algorithms we find the ratio NI/NC to make the comparison easy. We see that IsoRank is computationally more complex because it requires all vs. all protein similarities which are calculated using BLAST algorithm. This process is time consuming for large protein interaction networks. We also perform our experiments on the organisms *Caenorhabditis elegans, Drosophila melanogaster, Mus musculus*from the IsoRank dataset and species Flowering plant (*Arabidopsis thaliana*), House mouse (*Mus musculus*) and Street rat (*Rattus norvegicus*) from PINALOG dataset. We compare the performance of our algorithm and conclude that our algorithm produces alignment results that have more biological significance in comparison to IsoRank. These results can be used as a proof for the validity of our proposed method.

We know that network alignment results can be used for identifying conserved functional modules, predict protein functions, validate protein interactions, predict protein interactions or discover protein complexes. The results of the method described in this thesis can be used to predict protein complexes in the given species or predict the function of proteins by inheriting the annotation available of the aligned protein from the other species. In this thesis we limit ourselves to three species due to computational limitations. We consider the problem of aligning multiple networks as a multi-dimensional problem. Since multi-dimensional problem is said to be NP-Hard we would have to devise new heuristics to align more than three networks.

REFERENCES

Adamcsek, B., Palla, G., Farkas, I., Derényi, I., & Vicsek, T. (2006). CFinder: locating cliques and overlapping modules in biological networks. *Bioinformatics, 22*(8), 1021-1023.

Barabási, A., & Oltvai, Z. (2004). Network biology: understanding the cell's functional organization. *Nature Reviews Genetics, 5*(2), 101-113.

Brohee, S., & van Helden, J. (2006). Evaluation of clustering algorithms for protein-protein interaction networks. *BMC Bioinformatics, 7*(1), 488.

Bukard, R., & Cela, E. (1999). *Linear assignment problems and extensions.* Springer US.

Chen, L., Wang, R., & Zhang, X. (2009). *Biomolecular Networks: Methods and Applications in Systems Biology.* John Wiley and Sons.

Denièlou, Y., Boyer, F., Viari, A., & Sagot, M. (2009). Multiple alignment of biological networks: A flexible approach. In *Combinatorial Pattern Matching* (pp. 263-273). Springer.

Flannick, J., Novak, A., Do, C., Srinivasan, B., & Batzoglou, S. (2009). Automatic parameter learning for multiple local network alignment. *Journal of Computational Biology, 16*(8), 1001-1022.

Flannick, J., Novak, A., Srinivasan, B., & McAdams, H. (2006). Graemlin: General and robust alignment of multiple large interaction. *Genome Research, 16*(9), 1169-1181.

Fortunato, S. (2010). Community detection in graphs. *Physics Reports, 486*(3), 75-174.

Huang, G., & Lim, A. (2006). A hybrid genetic algorithm for the three-index assignment problem. *European Journal of Operational Research, 172*(1), 249-257.

Kalaev, M., Bafna, V., & Sharan, R. (2008). Fast and accurate alignment of multiple protein networks. In *Research in Computational Molecular Biology* (pp. 246-256). Springer-Verlag Berlin, Heidelberg 2008.

Kalaev, M., Smoot, M., Ideker, T., & Sharan, R. (2008). NetworkBLAST: Comparative analysis of protein networks. *Bioinformatics, 24*(4), 594-596.

Kelley, B., Sharan, R., Karp, R., Sittler, T., Root, D., Stockwell, B., et al. (2003). Conserved pathways within bacteria and yeast as revealed by global. *Proceedings of the the National Academy of Sciences of the United, 100*(20), 11394-11399.

Kelly, L., & Sternberg, M. (2009). Protein structure prediction on the Web: a case study using the Phyre server. *Nature protocols, 4*(3), 363-371.

Koyutùrk, M., Grama, A., & Szpankowski, W. (2005). Pairwise local alignment of protein interaction networks guided by. In *Research in Computational Molecular Biology* (pp. 48-65). Springer-Verlag Berlin, Heidelberg 2005.

Koyutùrk, M., Kim, Y., Topkara, U., Subramanian, S., Szpankowski, W., & Grama, A. (2006). Pairwise alignment of protein interaction networks. *Journal of Computational Biology, 13*(2), 182-199.

Kuchaiev, O., & Pržulj, N. (2011). Integrative network alignment reveals large regions of global network similarity in yeast and human. *Bioinformatics, 27*(10), 1390-1396.

Kuchaiev, O., Milenković, T., Memišević, V., Hayes, W., & Pržulj, N. (2010). Topological network alignment uncovers biological function and phylogeny. *Journal of the Royal Society Interface, 7*(50), 1341-1354.

Kuhn, H. (1955). The Hungarian method for the assignment problem. *Naval research logistics quarterly, 2*(1-2), 83-97.

Liao, C., Lu, K., Baym, M., Singh, R., & Berger, B. (2009). IsoRankN: Spectral methods for global alignment of multiple protein. *Bioinformatics, 25*(12), 253-258.

Lord, P., Stevens , R., Brass, A., & Goble, C. (2003). Investigating semantic similarity measures across the Gene Ontology: the relationship between sequence and annotation. *Bioinformatics, 19*(10), 1275-1283.

Narayanan, M., & Karp, R. (2007). Comparing protein interaction networks via a graph match-and-split algorithm. *Journal of Computational Biology, 14*(7), 892-907.

Özgen, C. (2007). Assignment problem and its variations (Doctoral dissertation, MIDDLE EAST TECHNICAL UNIVERSITY).

Palla, G., Derényi, I., Farkas, I., & Vicsek, T. (2005). Uncovering the overlapping community structure of complex networks in nature and society. *Nature, 435*(7043), 814-818.

Phan, H., & Sternberg, M. (2012). PINALOG: a novel approach to align protein interaction networks—implications for complex detection and function prediction. *Bioinformatics, 28*(9), 1239-1245.

Pierskalla, W. P. (1968). Letter to the Editor—The Multidimensional Assignment Problem. *Operations Research, 16*(2), 422-431.

Schlicker, A., Domingues, F., Rahnenführer, J., & Lengauer, T. (2006). A new measure for functional similarity of gene products based on Gene Ontology. *BMC Bioinformatics, 7*(1), 302.

Sharan, R., Suthram, S., Kelley, R., Kuhn, T., McCuine, S., Uetz, P., et al. (2005). Conserved patterns of protein interaction in multiple species. *Proceedings of the National Academy of Sciences of the United States of America, 102*(6), 1974-1979.

Singh, R., Xu, J., & Berger, B. (2007). Pairwise global alignment of protein interaction networks by matching. *Research in Computational Molecular Biology*, 16-31.

Singh, R., Xu, J., & Berger, B. (2008). Global alignment of multiple protein interaction networks with application to functional orthology. *Proceedings of the National Academy of Sciences, 105*(35), 12763-12768.

Tang, L., & Liu, H. (2010). Community detection and mining in social media. *Synthesis Lectures on Data Mining anf Knowledge Discovery*, *2*(1), 1-137.

Walhout, A., Sordella, R., Lu, X., Hartley, J., Temple, G., Brash, M., et al. (2000). Protein interaction mapping in C. elegans using proteins involved in vulval development. *Science, 287*(5450), 116-122.

Wang, J., Du, Z., Payattakool, R., Philip, S., & Chen, C. (2007). A new method to measure the semantic similarity of GO terms. *Bioinformatics, 23*(10), 1274-1281.

Wheeler, A., Barrett, T., Benson, D., Bryant, S., Canese, K., Chetverin, V., et al. (2007). Database resources of the national center for biotechnology information. *Nucleic acids research, 35*(suppl 1), D5-D12.

Yu, H., Luscombe, N., Lu, H., Zhu, X., Xia, Y., Han, J., et al. (2004). Annotation transfer between genomes: protein–protein interologs and protein–DNA regulogs. *Genome Research, 14*(6), 1107-1118.

VITA AUCTORIS

NAME:	Arushi Arora
PLACE OF BIRTH:	Ludhiana, PB, India
YEAR OF BIRTH:	1988
EDUCATION:	MSc. Computer Science (2011-2013)
	University of Windsor
	Windsor, ON, CA.
	B.E. Information Technology (2006-2010)
	Bharati Vidyapeeth University,
	Pune,MH, India.